Trabajo Fin de Grado:
Análisis e implementación de un sistema de agrupamiento automático de tweets

Tutor: Julio Villena Román
Autor: Fernando España García
Titulación: Grado en Ingeniería de Sistemas de Comunicaciones

11 de marzo de 2014

Agradecimientos

En primer lugar me gustaría agradecer a mi tutor Julio Villena su ayuda y dedicación, sin la cual este proyecto nunca habría sido posible. También me gustaría hacer extensible este agradecimiento a todos aquellos profesores que he tenido con un entusiasmo por el estudio que me han sabido transmitir.

Agradecer muy especialmente a mis padres, Montse y Vicente, que siempre que lo he necesitado han estado ahí, y sin los cuales no me habría sido posible llegar hasta aquí, así como a mi hermano Eduardo con el que he compartido toda una infancia. A mi abuela Josefa, que siempre se ha preocupado de todos y a mis abuelos que ya no están entre nosotros. A todos mis tíos y primos por todo su ánimo.

Y por supuesto no puedo olvidarme de todos mis amigos, de Miranda, Madrid y Paderborn, con los que he compartidos tantas experiencias, ilusiones, buenos y malos momentos, porque sé que siempre estaréis ahí, y sabéis que siempre estaré ahí.

Resumen

Hoy en día, y debido al auge de las tecnologías de la comunicación tenemos acceso a mucha mayor cantidad de información de la que podemos gestionar. Debido a esta problemática surge la necesidad de la minería de datos, esto es, de la gestión de la información de modo que se consiga extraer de la información recibida lo que realmente resulte interesante para unos determinados objetivos. Es en este campo en el que se encuadra el estudio de este trabajo. Para ser más exactos nos centramos en el campo del clustering, que es el agrupamiento de la información, en nuestro caso textos, según diferentes criterios, como en nuestro caso es la temática tratada, así como la creación de los grupos en los que se dividirá la información.

El trabajo que aquí se presenta analiza las posibilidades de la agrupación de textos, también conocida como minería de textos, desde los aspectos teórico y práctico. Para ello se divide en dos partes, la primera de ellas analiza el estado actual de la citada ciencia realizando un especial hincapié en el agrupamiento de documentos según su temática, para realizar un acercamiento a la segunda parte. Se busca, en cualquier caso, cubrir los principales aspectos del clustering resulten o no prácticos para el caso particular que se desarrollará en la segunda parte. Se desea realizar un análisis crítico de lo que ya está desarrollado a nivel teórico, no la creación de un nuevo modelo de clustering, aunque sí que se trate de buscar la mejor opción, así como de identificar y solventar los posibles problemas que se dan, dentro de cada aspecto y de cada rama del clustering.

La segunda parte consiste en el desarrollo práctico de un sistema capaz de dividir tweets en diferentes grupos según la temática de éstos. Se ha realizado la implementación en Java, y se ha elegido de entre los algoritmos estudiados en la primera parte el algoritmo llamado k-medias. También se ha implementado el algoritmo de herencia aglomerativa, pero los resultados mostrados por éste han llevado a rechazarlo como opción final en favor de la opción que emplea el algoritmo de k-medias, debido a una mayor tasa de error, y un tiempo de ejecución mucho mayor.

La implementación práctica ha servido como muestra de lo analizado en teoría, así como de análisis de los problemas que surgen en la práctica, y que a penas son estudiados en la teoría, pero que deben ser gestionados obligatoriamente cuando se desea llegar a una solución final. De este modo se ha conseguido ofrecer tanto una aproximación teórica como una práctica.

Palabras clave: Clustering, minería de datos, minería de textos, texto, tweet, Twitter.

Abstract

Nowadays, because of the rise of communication technologies, we have access to such a big amount of information that it becomes almost impossible to handle. Because of that, data mining has become necessary. Data mining is the management of information in order to get the part of it that is relevant for given objectives. This project focuses not on the whole of this field but on a specific area called clustering, meaning the grouping of information according to certain criteria,such as the making of groups to be filled with given information. Texts are grouped using the criteria of the topics of each of them.

This project analyses the possibilities of text grouping, also known as text clustering, covering both theoretical and practical aspects. In order to achieve this goal, this text is divided into two different parts. The first analyses the state of data mining, focusing on document grouping according to theme, in order to approach the second part of the project. All the main aspects of clustering are studied whether or not they are relevant to the second part of the project. The objective is to do a reasoned analysis of the aspects of data mining which are already developed in a theoretical way, not creating a new part of the given science, but trying to find the best option, such as identifying and solving the problems which could be found, in each aspect and part of the clustering.

The second part of the project is the practical development of a system which is able to divide tweets into different groups according to their theme. It has been implemented in Java, and the chosen algorithm of those studied is k-means. An agglomerative heritage algorithm has also been implemented, but according to the results it has been ultimately rejected. The two main grounds for this decision were the bigger error rate and the longer running time of the agglomerative heritage algorithm.

The practical implementation has been not only a good way to show the theme analysed in the theoretical part, but also a good way to find the problems which are ignored by the theory but become critical when a practical solution is expected. With this project both a theoretical and a practical approximation have been achieved.

Keywords: Clustering, data mining,text, text mining, Twitter, tweet

Índice

Índice de figuras

Índice de cuadros

Parte I
Presentación

1. Motivación

En nuestra sociedad tenemos la suerte de tener acceso a una cantidad de información que a duras penas podría ser imaginada en otros tiempos. Esta información proviene de múltiples fuentes, como pueden ser libros, revistas especializadas, periódicos... pero sobre todas ellas, hemos de destacar la gran revolución de nuestro tiempo, Internet. Internet es una fuente casi infinita de información, hasta tal punto, que en muchas ocasiones el usuario percibe tanta información, una gran parte de la cual no necesita, que no es capaz de acceder a la que sí le interesa. La presencia de información no deseada es denominada "ruido" dentro del estudio de la información. Ésta, junto con la ausencia de la información que sí se desea, denominada "silencio", hacen que la gran disponibilidad de información, pase de ser un lujo con grandes beneficios, a llegar a suponer un problema.

Es en este ámbito en el que se hace necesario un desarrollo de los métodos de búsqueda de esa información. Afortunadamente la gran capacidad de computación de los ordenadores, que ha ido creciendo más allá incluso de lo que la ley de Moore predijo [1] [1], nos permite el empleo de mayor número de algoritmos, así como de algoritmos más complejos cada vez. A su vez, se han ido desarrollando algoritmos de gestión de la información. Ahora, es menester emplear ambos recursos disponibles, con la finalidad de refinar la información ofrecida al usuario final.

[1] Moore predijo que la potencia de los procesadores se duplicaría cada 18 meses, y los resultados posteriores han venido a darle la razón.

2. Clustering: Clasificando conceptos

Uno de los principales problemas con los que se encuentra el usuario al navegar por Internet es que al existir información sobre temas tan variados, un mismo término puede aludir a conceptos temáticos diferentes.

Es ahí donde entra en escena el clustering, que no es otra cosa que el agrupamiento de elementos en virtud de sus parecidos y diferencias.

El clustering tiene validez más allá de la selección de información que nos interesa, siendo utilizado en ámbitos tan diversos como la biología, el marketing, la química... ([2] Capítulo: 1.1 Introduction)

Es en este punto donde debe realizarse una apreciación importante, puesto que suele conducir a error. El clustering no consiste solamente en la discriminación de elementos en diferentes grupos, sino que también consiste en la generación de esos mismos grupos. La simple división de los componentes en grupos previamente generados se denomina clasificación ([3]Capítulo: 7. Introduction) y en nuestro estudio resulta insuficiente, puesto que, como ya se ha comentado, la información en Internet abarca ámbitos tán diversos que los grupos a generar en diferentes búsquedas no tienen por qué tener nada que ver entre sí. Entiéndase que si bien en cada una de las búsquedas puedan estar claros los grupos en los que han de ser divididos los resultados, el algoritmo debe estar preparado para todas ellas, por lo que resulta implanteable que disponga a priori de estos grupos, además de que un algoritmo que haga uso de grupos predefinidos tendría el problema de quedarse obsoleto con gran velocidad, debido a que la información de Internet es bastante dinámica. A este respecto es importante comentar que si bien se está estudiando el clustering, los ejemplos empleados en este texto aluden a casos de clasificación, al menos muchos de ellos, puesto que complicaría bastante la explicación el uso de conjuntos desconocidos, ya que no podrían resultar, por definición, conocidos al usuario.

También hemos de ser conscientes de que el clustering forma parte de una parte del tratamiento de los datos mucho más amplia, que se denomina minería de datos (data mining). La minería de datos trata de obtener patrones y conocimiento a partir de un conjunto de datos dado ([4] Capítulo: 1.2). La comprensión de la minería de datos puede llevarnos a entender mejor tanto el funcionamiento como los límites del clustering.

3. Minería de datos

En primer lugar deberíamos aclarar la diferencia entre dos conceptos que si bien guardan cierta relación entre sí, deben estar claramente separados. Estos conceptos son el de dato y el de conocimiento. Ésta es la definición que nos da la Real Academia Española de la Lengua sobre la palabra dato:

> R.A.E.:
> "dato1.
> (Del lat. datum, lo que se da).
> 1. m. Antecedente necesario para llegar al conocimiento exacto de algo o para deducir las consecuencias legítimas de un hecho."

Como muy bien nos indica la R.A.E., un dato es la base de donde puede obtenerse un conocimiento, pero no el conocimiento en sí. Es a partir de diferentes datos, de donde, si estos son tratados convenientemente, puede llegarse a obtener un cierto conocimiento. Pero es importante reseñar que ha de darse ese tratamiento para que el conocimiento se obtenga.

Una metáfora bastante gráfica de esta situación es la minería. Dentro de una montaña hay muchos minerales, por ejemplo oro. Pero ha de picarse la montaña, tratar las piedras resultantes, y separar el oro de la ganga para poder disponer de las pepitas que todos conocemos. Es de esa similitud de la que nace el concepto minería de los datos. Hemos de buscar de entre todos los datos disponibles aquellos que guarden relación con lo que estamos buscando, y posteriormente hemos de depurarlos, ordenarlos y presentarlos correctamente para obtener el citado conocimiento.

De un modo más preciso podríamos definir la minería de datos como el acto de obtener información de los datos que no era perceptible originalmente, y que resulta fundamental para el objeto de estudio. Es ahí donde entra en escena el clustering, ya que nos permite descubrir grupos de elementos a través de similaridades entre ellos que no habían sido percibidas anteriormente, debido a lo cual debe ser el propio clustering el que genere estos grupos, ocultos a nuestra percepción.

En este punto surge otro problema, este proceso tiene múltiples pasos. Dándose el caso de que algunas veces se denomina minería de datos a todo el conjunto, mientras que otras se denomina minería de los datos a uno solo de los pasos, como puede observarse en la figura ([4]Capítulo: 1.2):

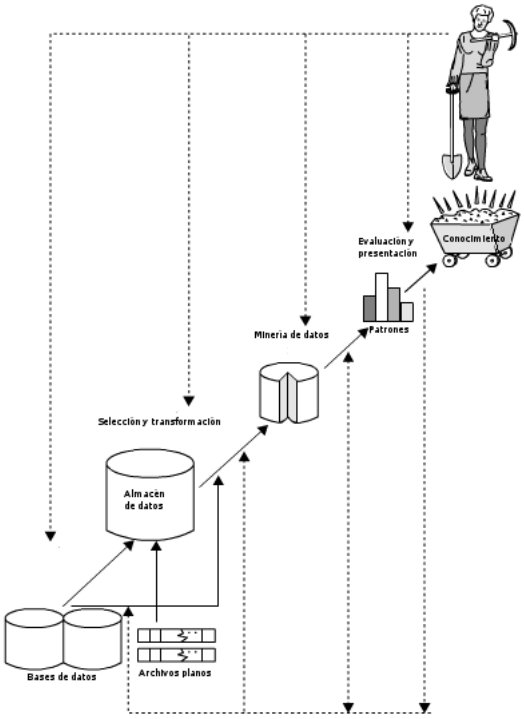

Figura 1: Imagen que representa el proceso completo por el que pasan los datos, extraida del libro "Data mining : concepts and techniques" ([4] Capítulo: 1.2)

Al respecto hemos de especificar que el clustering es una de las opciones de la minería de datos en sentido estricto. Aunque esto implica, de modo trivial, que también pertenezca a la minería de datos en sentido amplio.

La minería de datos en sentido amplio incluye más pasos, que Sang C. Suh, en su libro "Practical Applications of Data Mining" describe como sigue ([4] Capítulo: 1.2):

- limpieza , que no es otra cosa que el borrado del ruido, así como de los datos inconsistentes.
- Integración, o unificación de datos provenientes de varias bases de datos.
- Selección , es decir, extracción de los datos relevantes.
- Transformación, entiéndase como modificación del formato de los datos para hacerlos adecuados a la minería.
- Minería, la susodicha minería de datos en sentido estricto.
- Evaluación de patrones, identificado de los patrones realmente interesantes para el conocimiento buscado.

- Presentación del conocimiento, tratamiento del conocimiento ya obtenido para mostrarlo a los usuarios.

Hemos de tener en cuenta que nosotros trabajamos con un conjunto de tweets, por lo que ya se han producido los pasos anteriores a la minería. Debido a esta razón nos centramos simplemente en la minería, y además no en toda ella, solamente en la opción del clustering.

4. Marco regulador

Los potenciales problemas legislativos del problema a encarar se enmarcan dentro del ámbito de la gestión de información, al tratarse de un programa que divide tweets, cuyos autores no tienen necesariamente que tener relación con los desarrolladores de la aplicación.

En este sentido ha de hacerse una división importante. El programa en cuestión, a pesar de estar diseñado para trabajar con tweets, no contiene ningún tweet, ni información personal, de carácter privado o público, de ningún usuario de la aplicación, ni de Twitter. Donde sí que hay tweets así como información de sus autores es en los ficheros empleados en testeo y calibración. Por tanto es ahí donde ha de realizarse un análisis de la legislación vigente con la finalidad de no incurrir en ningún tipo de delito ni falta.

Con esta finalidad se realiza una consulta en la página web de la Agencia Española de Protección de Datos (http://www.agpd.es), en la que se consulta la legislación vigente en el territorio nacional en el momento de elaboración de la aplicación. En primer lugar se consulta la ley sobre protección de datos [9] , de aplicación en la actualidad en todo el territorio español. En el artículo tercero del título primero de la citada ley puede leerse la definición de "datos de carácter personal", que se muestra a continuación:

> Datos de carácter personal: cualquier información concerniente a personas físicas identificadas o identificables.

Como puede leerse, para que un dato resulte de carácter personal ha de resultar posible la identificación del aludido. Este hecho nos lleva a preguntarnos si realmente es el caso de un tweet, o de alguno de sus metadatos, incluidos también en el fichero. Se identifican como metadatos que han de ser tenidos en cuenta en esta valoración los siguientes: "name", "user", "description", "image","geo", "coordinates", "place" y "text".

El caso de los metadatos "geo", "coordinates" y "place" puede ser tratado en común, ya que todos ellos tratan de determinar la localización del usuario en el momento en el que se envió el tweet. Estos metadatos sí que podrían ser considerados de carácter personal, ya que al emplearse en conjunto sobre un mismo usuario, podrían dar conocimiento sobre sus rutinas, incluyendo lugar de trabajo y/o estudios, lo que, además, podría facilitar la identificación del afectado, siempre que se cuente con más datos, puesto que por sí mismos resultarían claramente insuficientes. Ha de apostillarse, eso sí, que el usuario puede decidir si incluye la información sobre su localización o no lo da, opción que se muestra en "Seguridad y Privacidad", en la segunda sección, la de "Privacidad".

El caso de los datos metadatos "name", "user" e "image" también puede tratarse como uno solo. En este caso la importancia de los datos es relativa, puesto que depende de las decisiones del usuario en cuanto a qué datos aportar. Hay usuarios que emplean como nombre de usuario ("name") su propio nombre, en algunos casos incluso con sus dos apellidos, o un subconjunto de estos, resultando entonces los aludidos como identificados. Por otra parte hay usuarios

que emplean sobrenombres que nada tienen que ver con sus nombres oficiales, con lo que en este segundo caso la identificación no resultaría posible. En el tema de la imagen sucede lo mismo. Por una parte contamos con usuarios que ponen como imagen una fotografía suya, o incluso suya en compañía de un amigo o familiar, caso en el que la identificación sería mucho más que factible, frente a otros usuarios que ponen imágenes que nada tienen que ver con ellos, excepto el hecho de gustarles, resultando en este caso en una identificación casi imposible. En cualquier caso ha de tenerse en cuenta que son datos aportados voluntariamente por el usuario, en conocimiento de que aparecen a todo aquel que acceda a su perfil de Twitter. A pesar de la voluntad del usuario de hacer públicos los datos, los consideraremos personales por el caso en el que aluden al usuario directamente, el caso peor que hemos citado.

Todos estos metadatos analizados hasta el momento tienen una ventaja, y es la de que no son empleados por el programa, desafortunadamente no se puede decir lo mismo del último caso enumerado, el campo "text", este campo es el que contiene el tweet en sí, y conjuntamente con el campo "id" los únicos que son extraídos de los ficheros de tweets para ser usados por el programa, así como mostrados al final. El campo "text" ha de ser necesariamente empleado en el análisis, puesto que es el campo empleado durante todo el análisis, el único que contiene el mensaje del usuario, y por lo tanto toda su temática. En cuanto al campo "id", no contiene información susceptible de permitir la identificación del usuario, ya que es un número único asociado a la cuenta, pero solo a la cuenta, por lo que no aporta información sobre la persona que está detrás de la misma. El campo "text" nos vuelve a plantear la misma situación que los campos "name", "user" e "imagen", ya que cabe la posibilidad de que el usuario esté hablando sobre su vida personal, y/o aportando datos que permitan identificarle, pero también puede estar hablando de un tema que no tenga que ver con su persona, como pueden ser, de forma no excluyente, un suceso de carácter público o una divulgación científica.

Continuando en el artículo tercero volvemos a encontrarnos con una descripción que podría ser de interés para nuestro problema particular, es la descripción del procedimiento de disociación, que dice como sigue:

> Procedimiento de disociación: todo tratamiento de datos personales de modo que la información que se obtenga no pueda asociarse a persona identificada o identificable.

Esta descripción nos permite reducir el número de datos potencialmente peligrosos, o al menos la influencia de ellos. Ha de tenerse en cuenta que el programa solo carga los datos asociados a los campos "id" y "text", podría deducirse que se está realizando un proceso de disociación del resto de datos que se han estudiado más arriba, con lo que el problema asociado a estos datos se ciñe a la existencia de los ficheros en sí, y no a los resultados mostrados por el programa. Continua, eso sí, la problemática asociada al campo "text", ya que este campo sí que es empleado por el programa.

En el título segundo artículo cuarto punto segundo de la citada ley se indica claramente como el análisis estadístico de los datos sí que está protegido por la

propia ley,como podemos observar a continuación:

> Los datos de carácter personal objeto de tratamiento no podrán usarse para finalidades incompatibles con aquellas para las que los datos hubieran sido recogidos. No se considerará incompatible el tratamiento posterior de éstos con fines históricos, estadísticos o científicos.

Este hecho eximiría de cualquier atentado contra la citada ley el uso que se hace en la presente memoria. Los propios resultados del programa también estarían protegidos siempre que no sea empleado con fines comerciales, ya que podrían ser considerados un estudio de finalidad científica.

Continuamos el análisis de la ley en busca de la legalidad o ilegalidad de los casos en los que todavía no se ha encontrado base legal, lo que nos lleva al mismo título, artículo seis punto segundo, que mostramos a continuación:

> No será preciso el consentimiento cuando los datos de carácter personal [...] cuando los datos figuren en fuentes accesibles al público y su tratamiento sea necesario para la satisfacción del interés legítimo perseguido por el responsable del fichero o por el del tercero a quien se comuniquen los datos, siempre que no se vulneren los derechos y libertades fundamentales del interesado.

Twitter es una fuente accesible al público debido a que todo aquel con acceso a Internet puede entrar en la página y leer los mensajes escritos por los usuarios en abierto, como son todos los contenidos en los ficheros empleados, con lo que el campo "text" puede ser considerado como figurante en una fuente accesible al público. Este hecho, junto a la disociación del resto de campos problemáticos nos lleva a deducir que se ha respetado la legalidad vigente en el país en todo momento.

5. Objetivos

En este Trabajo Final de Grado se va a realizar una investigación de los algoritmos de clustering orientados a la clasificación de tweets, posteriormente se van a elegir e implementar dos posibles soluciones, y por último se van a analizar y comparar determinando cual es la óptima.

En la implementación se requiere que el programa sea capaz de dividir tweets que se le introduzcan por temática en grupos mostrados a la salida. Se requiere, también, que el número de grupos mostrados a la salida pueda ser determinado por el usuario.

Parte II
Estado del arte

6. Introducción

Son muchos los algoritmos de clustering, que emplean métodos muy diferentes entre sí para obtener los buscados grupos de elementos. No es que unos sean intrínsecamente mejores que otros, sino que cada uno es el mejor, de entre ellos, para una finalidad en particular. esto se debe al hecho de que no existe un criterio objetivo transversal a todo el clustering, qué nos permita determinar que grupos son mejores que otros. Por el contrario, debe ser el análisis de cada ciencia en particular el que determine qué grupos son más prácticos para sus fines.

Los algoritmos de clustering se dividen en conjuntos, según su percepción de lo que es un grupo, y según diferentes criterios de lo que es un grupo, o cluster. La comprensión de estos criterios es de suma importancia para comprender la filosofía que hay detrás de cada conjunto de clusters, y poder decidir de forma razonada cual es el mejor para cada finalidad.

7. ¿Qué es un grupo?

> R.A.E.:
> "grupo.
> (Del it. gruppo).
> 1. m. Pluralidad de seres o cosas que forman un conjunto, material o mentalmente considerado."

Si bien existe un acuerdo general respecto a lo que es un grupo, no hay un acuerdo tan universal respecto a ciertas normas que debe cumplir, o no, cada tipo de grupo. Estas diferencias aluden más a los intereses de cada ámbito en el que se emplea el clustering, que a razones intrínsecas de comprensión del concepto. Pasamos a describir los aspectos más importantes en los que se difiere a la hora de comprender un concepto.

7.1. Herencia

La herencia en este ámbito se define como la pertenencia de un conjunto a otro conjunto mayor. Éstos conjuntos más grandes pueden contener tanto a otros conjuntos más pequeños y a elementos sueltos, es decir, que no pertenezcan a ningún conjunto menor, como solamente a otros conjuntos menores. A su vez podría producirse este hecho de modo recursivo, dándose el caso de grupos, que contengan a grupos, que a su vez contengan a otros grupos, y así sucesivamente. Eso sí, siempre de mayor a menor, es decir, un grupo no podría contener a un "superior".

La existencia o no de herencia es uno de los puntos a definir en los conjuntos, puesto que ambas opciones son válidas. Ha de determinarse en cada caso la mejor opción en función de las necesidades del campo de estudio.

Un ejemplo clásico de conjuntos con herencia es la biología, en la que se van realizando divisiones progresivas de las formas de vida, entiéndase, varias razas pertenecen a la misma especie, que a su vez pertenece a un conjunto y así sucesivamente. Éste sería además un buen ejemplo de herencia en la que un elemento no puede pertenecer a un grupo superior sin pertenecer a cada uno de los inferiores, puesto que todo individuo está clasificado en cada uno de los subgrupos desde el más pequeño hasta el más grande.

Por último hemos de tener en cuenta la posibilidad de que no exista la herencia en un tipo de grupos en particular. Un buen ejemplo serían los tipos de estrellas. Existen diferentes tipos de estrellas, según su edad, el análisis de sus características nos podría dar lugar a un clustering en el cual toda estrella ha de pertenecer a un grupo, pero nunca a ningún subgrupo, puesto que no se considera ningún subgrupo respecto a las características de una estrella dada.

Éste ha de ser un aspecto a analizar en la selección del algoritmo, ya que podría tener sentido tanto la existencia de herencia, como su no existencia. Teniendo ambas puntos a favor y puntos en contra. Siendo básicamente los puntos a favor de una los puntos en contra de la opuesta, y viceversa. Detallamos en primer lugar los puntos a favor de la existencia de herencia:

- Reducción de los grupos mostrados al usuario en un momento dado.
- Capacidad de mayor precisión en cuanto a seleccionar la temática deseada.
- Posibilidad de acceder con sencillez a temáticas cercanas a la deseada para extender conocimiento.

A continuación detallaremos los puntos a favor que tendría la no existencia de herencia:

- Mayor rapidez de acceso al grupo deseado, al realizarse en un solo paso.
- Simplicidad de manejo, al no tener el usuario que moverse por varias capas.
- Posibilidad de mostrar los resultados por "pestañas".

Estos puntos a favor y en contra deben ser tenidos en cuenta al realizar la selección final, valorando de modo subjetivo el peso que se le asigne a cada uno.

7.2. Fuzzy clustering

El término fuzzy clustering podría ser traducido al castellano como difuso o borroso, aunque sería más claro traducir como clustering relativo. En contraposición encontramos el hard clustering, que traduciríamos literalmente como clustering duro, pero que, sin embargo, sería más comprensible traducir como clustering absoluto.

En este punto nos preguntamos, ¿Qué supone que un clustering sea relativo o absoluto? La respuesta es bien sencilla. Un elemento dado puede pertenecer a un grupo, o a varios, de modo completo, o con una cierta probabilidad. Es decir, puede entenderse que pertenece a un grupo o no, por una parte, o que pertenece a cierto grupo al "x por ciento". El primer caso, pertenecer "sí o no" a un grupo es el denominado como clustering absoluto, y el segundo, la pertenencia en un tanto por ciento a un grupo, es el llamado clustering relativo.

Otra vez vuelve a darse el caso de que no puede considerarse de modo absoluto uno mejor que el otro, sino que cada caso puede ser mejor para un estudio en particular. Por poner un ejemplo el custering relativo sería una buena opción en el caso de estudiar la pertenencia de un grupo de perros a una raza en particular. Hay perros que pertenecen a una sola raza, y así tienen un pedigrí que lo acredita, pero ese no es el caso más habitual. Sino que el caso más habitual es que un perro tenga genética de varias razas. Podríamos considerar aún así la pertenencia de cada perro de forma booleana a cada una de estas razas, diciendo que el perro X pertenece a las razas Y y Z, pero eso nos aportaría una información bastante pobre sobre el animal, ya que hay veces en las que dos perros tienen componentes de las mismas razas, pero características completamente diferentes, debido al hecho de que uno de ellos tiene la mayoría de su genética de una de las razas, mientras que el otro la tiene del otro. Debido a este hecho es mucho más recomendable el uso del clustering relativo, en el que podríamos decir que el primer perro tiene un 10 % de la raza Y, y un 90 % de

la raza Z, mientras que el segundo tiene un 90 %, de la raza Y y un 10 % de la raza Z.

Por el contrario nos encontramos con otras situaciones en las que el clustering relativo no tiene ningún sentido. Ésta situación se dá cuando un elemento puede solamente pertenecer a un grupo o no, no pudiéndose concebir una cuantificación de la pertenencia. Un buen ejemplo podrían ser las nacionalidades. Si bien una persona puede tener más o menos vínculos con un país así como características en común con el grueso de su población, la nacionalidad es algo absoluto, se tiene o no se tiene. La división de personas respecto a su nacionalidad, puede darnos individuos que pertenezcan a un grupo, la mayoría, a varios, dobles nacionalidades, o a ninguno, apátridas, pero en los tres casos de modo absoluto.

Hay muchos casos, como puede ser el de las razas de perros mostrado anteriormente, en el que puede decidirse entre ambos tipos de clustering. La elección del clustering absoluto suele llevarnos a un resultado más sencillo, mientras que la elección del clustering relativo suele llevarnos a resultados más completos, ya que no hemos desechado una cierta información. En cada caso debe tomarse la decisión entre ambas opciones, valorando si la cantidad de información perdida por el clustering absoluto merece la pena o no en relación a la simplificación que sufre el problema.

En otros casos puede darse que el clustering relativo carezca de sentido en relación con la presentación del conocimiento obtenido, o el uso, de modo más general, que desea hacerse de éste.

El clustering relativo podría ser de mucha utilidad en el estudio de cada individuo particular, en el que se puede mostrar mediante gráficos de barras, circulares, o cualquier otro que represente componentes porcentuales. También podría ser de utilidad en estudios estadísticos, en los que mostrar, por ejemplo, la pertenencia media de unos elementos a un grupo dado.

El clustering absoluto tiene más sentido, cuando se trata de estudiar, de forma no estadística, a grupos enteros, ya que es difícil realizar muchos estudios cuando ha de tenerse en cuenta la pertenencia relativa a un grupo. Por ejemplo es difícil saber con cuantos elementos puede contar un grupo, ya que habría que considerar si las pertenencias muy bajas han de ser tenidas en cuenta o no, o bien si se cuenta a los individuos como fraccionarios en función de su porcentaje de pertenencia. Una forma de adaptar el clustering relativo a este tipo de situaciones sería convertirlo en clustering absoluto considerando un porcentaje dado de pertenencia a partir del cual un elemento se pasa a considerar sencillamente como que pertenece al grupo, y por debajo del cual se considera que simplemente el elemento no pertenece al grupo. Un ejemplo de esta situación es la nota que se asigna a un alumno. La valoración de sus exámenes, trabajos, participación... da lugar a una valoración numérica, que podríamos considerar como la pertenencia relativa al grupo de los que han adquirido los conocimientos ofrecidos en un curso dado. Pero cuando la evaluación finaliza, esta especie de clustering relativo ha de ser convertida en clustering absoluto para determinar, no ya en que medida ha adquirido el alumno los conocimientos, sino si el alumno está aprobado o suspendido. El caso más habitual, aunque no el único, es que se tome de frontera el 50 %, a partir del cual el alumno es asignado al grupo

de los aprobados, y por debajo del cual el alumno es asignado al grupo de los suspendidos.

Al igual que en el caso anterior, hemos de estudiar cuál de las dos opciones sería mejor para nuestro problema. Debido a que lo que tratamos de hacer es mostrar las soluciónes divididas por grupos según temática, hemos de considerar la imposibilidad de hacer esto sin disponer de un clustering absoluto, ya que hemos de decidir en cada caso si un elemento pertenece, o no, a un grupo dado, con la finalidad de poder decidir si mostrarlo en esa sección.

Esto no implica necesariamente que no pueda hacerse, como paso intermedio, un clustering relativo. Estaríamos ante el caso mostrado anteriormente, en el cual se realiza inicialmente un clustering relativo, que posteriormente es truncado para conseguir un clustering absoluto. La pregunta que debemos hacernos es ¿Qué puede aportarnos la realización de este paso intermedio? Así como ¿Merece esa aportación el coste computacional que implica este clustering añadido? En cuanto a la primera pregunta podría aportarnos información extra para el usuario, como podría ser la de mostrar, dentro de los elementos qué finalmente han sido asignados a cada grupo, con que porcentaje pertenecen a este grupo, de modo que el usuario pueda valorar hasta que punto le interesa. Además este porcentaje podría servir al propio programa para reordenar los resultados, no ya en el orden ofrecido por el buscador, sino de mayor a menor pertenencia a cada grupo, mejorando el acceso del usuario a la información que más desea.

En cuanto al coste computacional derivado de tal decisión, debe ser estudiado en cada caso particular, por lo que, al igual que con la herencia, se deja a función de valorarlo en el estudio de cada algoritmo. Una ventaja es que en algunos casos es información que ya ha sido empleada por el programa en la generación del clustering absoluto, con lo que no se trataría de realizar muchas más operaciones, sino simplemente de pasar a mostrar datos que ya han sido obtenidos, así como de emplearlos para la reorganización citada. esto también nos lleva a rechazar el uso del clustering relativo en los algoritmos que directamente empleen un clustering absoluto, puesto que tendría que realizarse un cálculo aún más complejo que el original con la finalidad de obtener las ventajas citadas aquí.

Una de las posibles aplicaciones del uso del clustering relativo como paso intermedio es la de poder mostrar al usuario el porcentaje de pertenencia de cada elemento al grupo en el que está siendo mostrado como información añadida a la propia búsqueda. . Un posible punto negativo de esta opción es el aumento de la densidad de información ofrecida al usuario, que reduciría la claridad de lo mostrado. En cualquier caso ésta información es muy reducida (un simple porcentaje), por lo que no saturaría al usuario en exceso, no resultando grave el problema, a no ser que se acumule con otra mucha información añadida, situación en la que habría que elegir las informaciones añadidas más interesantes.

7.3. Exclusividad

Un aspecto al que ya hemos hecho alusión anteriormente, si bien todavía no nos hemos detenido con precisión a analizar, es la exclusividad. esto es, la pertenencia de un elemento a como máximo un grupo, dentro de un mismo nivel. El caso contrario sería el de la superposición, en el cual un elemento puede pertenecer a varios grupos del mismo nivel, con lo que estos grupos estarían superponiéndose entre sí.

En un clustering con exclusividad puede darse el caso de que un elemento no pertenezca a ningún grupo o que pertenezca a un grupo, pero nunca podría darse el caso de un elemento que pertenezca a varios grupos a la vez. Este tipo de clustering sería el más adecuado en situaciones en las cuales los grupos son excluyentes entre sí, resultando inconcebible la pertenencia a varios.

En un clustering sin exclusividad, es decir con superposición, un mismo elemento puede pertenecer tanto a ningún grupo, como a uno solo, a dos o incluso a más. Este caso es el que se daría en algunos de los ejemplos citados anteriormente. Es un caso muy habitual en el clustering relativo, ya que al pertenecer parcialmente a un grupo es natural, aunque no necesario, que el elemento pertenezca también a otros grupos.

No debe confundirse la superposición con la herencia, puesto que un elemento puede pertenecer a un grupo y a alguno de sus subgrupos, pero no formar parte de varios grupos en el mismo nivel. Un ejemplo, que además ya ha sido citado anteriormente, y que podría resultar bastante interesante en este caso es el de las divisiones de las formas de vida. En este caso un elemento dado pertenece a un grupo que es el de su especie, a otro superior que es el de su filo... y así hasta completar un gran número de grupos, debido a las múltiples divisiones que se realizan en la biología. Pero en ningún caso resultaría concebible que un ser vivo perteneciera a varias especies a la vez (téngase en cuenta que por debajo de la especie sí que pueden encontrarse seres que pertenezcan a varios grupos, como el ya citado caso de las razas de perros, pero nunca por encima). En este caso consideraríamos que estamos ante un grupo con herencia y con exclusividad, como conceptos separados. Un criterio de exclusividad más estricto podría ser el de exigir que se pertenezca a un solo grupo en total, independientemente del nivel, con lo que se estaría anulando la posibilidad de tener herencia, pero este criterio reduciría la información conjunta que nos aportan ambos datos, exclusividad y herencia, ya que todo grupo con herencia tendría automáticamente superposición, razón por la cual empleamos el criterio descrito anteriormente.

Un grupo con superposición sería por ejemplo el de las nacionalidades, que también hemos citado anteriormente. Un mismo individuo puede ostentar la nacionalidad de ningún país, de uno, o de varios. En este caso carecería de sentido dividir a los individuos forzando a que pertenezcan a uno solo de los grupos, por estar obligando a que parte de los individuos pierdan una parte vital de su información.

Tanto la superposición como la exclusividad tienen características que las hacen mejores para unos objetivos del clustering y peores para otros, por lo que

debe ser el análisis de cada situación el que nos haga decidirnos por una de ellas en particular. En primer lugar analizaremos las ventajas que aporta cada una, que serían las desventajas de la otra, para poder decidir con posterioridad en nuestro caso en particular.

Ventajas de la exclusividad:

- Menor información necesaria para cada individuo.
- Mayor simplicidad en la gestión de los grupos, al no tener individuos que compartir.
- No existe el aumento de datos por duplicidades, que la superposición produce al trabajar con los grupos por separado.

Ventajas de la superposición:

- Información más completa sobre la situación de cada individuo.
- No existe la necesidad de eliminar de un grupo a integrantes coherentes porque tengan más sentido en otro grupo.

Teniendo en cuenta las ventajas e inconvenientes generales de cada grupo hemos de hacer una particularización para nuestro caso, en el que unas ventajas pueden pesar más que otras, o incluso algunas pueden resultar carentes de toda importancia. Las ventajas más importantes son por una parte la no existencia de duplicidades de la exclusividad, y por otra la falta de necesidad de eliminar de un grupo a integrantes coherentes de la superposición. En este caso volvemos a encontrarnos con una situación en la que básicamente estamos enfrentando un aumento en la calidad de los resultados frente a un mayor coste computacional.

La combinación entre superposición y clustering relativo como paso intermedio aplicada a nuestro problema implicaría una sola especificación en el paso del clustering relativo al absoluto. Puesto que se ha de poner un porcentaje mínimo a partir del cual el elemento pertenece a un grupo, todos los grupos en los cuales el elemento tenga a partir de cierto porcentaje el elemento pertenecerá a ellos. Ha de entenderse que el número de grupos a los que pertenecerá un elemento dado depende de cual sea el porcentaje mínimo que determinemos. Tenemos dos modos de determinar ese porcentaje.

El primer modo es empírico durante el diseño. Éste consiste en hacer pruebas con búsquedas reales de modo que se obtenga un porcentaje que dé un número de grupos por elemento ni demasiado alto ni demasiado bajo en función de un criterio subjetivo de diseño. Este método nos arroja un problema consistente en que los elementos demasiado divididos entre múltiples grupos podrían no llegar a entrar en ninguno de ellos. Si bien es cierto que este es un problema muy relativo, puesto que un elemento que no tenga un porcentaje aceptable en ningun grupo quizá es que no deba pertenecer a ningún grupo. En cualquier caso la posibilidad de que un elemento pueda no pertenecer a ningún grupo será tratada en la sección siguiente, en la que analizaremos ventajas y desventajas de la misma.

El segundo modo de obtener este porcentaje se realizaría en cada búsqueda, y consistiría en determinar durante el diseño el número de grupos al que se desea que pertenezca cada elemento, eligiéndose en cada búsqueda que el elemento pertenezca a los "x" grupos con mayor porcentaje. De hecho en esta opción no se estaría considerando realmente una frontera como tal. El problema con el que nos podemos encontrar al emplear este segundo método se daría cuando un elemento tenga un porcentaje aceptable solo de un número de grupos menor al determinado, en este caso se estaría forzando a que sea mostrado en grupos con los que guarda menos relación de la deseada.

Una vez más, existe una tercera opción que es la combinación de las anteriores, consistiría en determinar un número máximo de grupos a los que un elemento ha de pertenecer y, además, un porcentaje mínimo de cada grupo que este elemento ha de tener para pertenecer a él. Ésta opción podría ser una solución al citado problema de la segunda opción, siendo más una mejora de esa opción, que una opción intermedia, debido al hecho de nos planteamos si la situación opuesta a la planteada en el primer caso, que es la de que un elemento pertenezca a todos o casi todos los grupos llega a ser un problema si realmente guarda una relación importante con cada uno de ellos.

Debido a los problemas arrojados por los otros casos optaremos por emplear la primera solución, a no ser que la selección empírica de un porcentaje mínimo se torne problemática, situación en la cual se pasaría a emplear la tercera opción, puesto que al ser la versión mejorada de la segunda sería mejor que ésta. Siempre en el caso de decidirnos por la superposición y no por la exclusividad, hemos de entender.

7.4. Clustering completo

El concepto de clustering completo alude al clustering en el cual se fuerza a que todo elemento estudiado pertenezca como mínimo a un grupo. La opción opuesta es el clustering parcial que es el clustering en el que se permite que un elemento pueda no pertenecer a ningún grupo.

En los diferentes ámbitos de aplicación del clustering en la vida real podemos encontrarnos con ambas situaciones, dependiendo de las necesidades de cada caso en particular. Un ejemplo ya citado en el que podemos observar un clustering completo es la división de animales en diferentes especies. Puede darse el caso de que un animal no pertenezca a ninguna de las especies conocidas hasta ser descubierto, situación en la cual se definiría una nueva especie, pues resulta inconcebible que un animal no pertenezca a ninguna especie en absoluto. Debido a esta situación un algoritmo de clustering aplicado a la separación animales entre diferentes especies no debería en ningún caso incluir la posibilidad de dejar sin especie a alguno de los animales analizados.

También existen situaciones en las cuales un elemento puede no pertenecer a ningún grupo, siendo esto completamente aceptable en esa situación. Un buen ejemplo que también ha salido anteriormente es la división de personas por nacionalidades, en la cual los citados apátridas serían el caso en el que un elemento no pertenece a ningún grupo. En este caso, lo que resultaría carente

de sentido es forzar a que estas personas pertenezcan a alguna nacionalidad, ya que se estaría negando su realidad. En este caso el algoritmo empleado debería permitir que un elemento quede sin grupo si en el estudio de ese elemento se considera que es lo más acertado.

Cuando tratamos de estudiar esta situación en nuestro algoritmo nos encontramos con lo que de forma estricta debería ser considerado un clustering completo, debido a que todo Twitter qué tenga algún sentido trata sobre algo, por lo que debería ser incluido en un grupo. Pero el mundo de Internet es tan amplio en los temas que trata, que podría darse el caso de que el número de grupos totales generados sea muy alto, reduciendo la claridad de los resultados mostrados, debido a que el usuario tenga ante sí más grupos de los que puede abarcar mentalmente.

Para que este hecho no se dé, una posible solución sería limitar el número total de grupos que se pueden generar, lo que no debe confundirse con el número total de grupos al que un elemento puede pertenecer. De este modo estaríamos convirtiendo nuestra solución en una de clustering parcial, pues habría elementos que no serían incluidos en ninguno de los grupos creados. Con la finalidad de que estos elementos sean mostrados a pesar de no tener grupo se podría generar un grupo final que se podría denominar "varios", "otros", "miscelanea"... En este grupo se mostrarían los elementos que no se han considerado suficientemente relevantes para ninguno de los demás grupos.

Ésta situación podría darse o no dependiendo de las características del algoritmo empleado. Si se considera que un elemento pertenece simplemente al grupo con el que guarda más relación, ya sea porcentualmente o por distancia, todo elemento va a pertenecer a un grupo, con lo que estaríamos ante un clustering completo, en el que no tendríamos que crear el grupo final de "varios". Por el contrario si se considera, como se ha citado anteriormente que un elemento pertenece a todo grupo con el que guarda una relación que expresada en porcentaje no baja de una cierta cantidad, sí que podría darse la situación de que un elemento no pertenezca a ningún grupo. En este caso sí que sería necesario el grupo "varios". Podría considerarse que debido a que el porcentaje requerido no sea demasiado alto la probabilidad de que esto suceda sea baja, pero el que el algoritmo conciba la posibilidad aumentaría su robustez, razón por la cual debe ser definida, puesto que el coste computacional de realizar esta comprobación no sería excesivo.

Otra posible solución sería la de generar el grupo "varios", pero no introducir en él directamente los resultados que no pertenecen a ninguno de los otros grupos mostrados, sino introducir en él todos los grupos que no han sido mostrados directamente por ser demasiado minoritarios. Ésta especie de falsa herencia haría que los resultados mostrados al usuario sean tan sencillos como en el empleo anterior del grupo "varios", siempre que el usuario no entre en la sección varios. Pero a la vez se reduciría el problema de haber perdido la información de a qué grupos pertenecen esos resultados que no pertenecen a los grupos mayoritarios. La problemática que tendría esta opción es la posible dificultad, según como se muestren finalmente los resultados, de mostrar la estructura de herencia al usuario. Dos ejemplos muy habituales son los de carpetas y los de

pestañas. En el ejemplo de carpetas la estructura de falsa herencia que estamos generando no daría el más mínimo tipo de problema, pero en la estructura de pestañas resultaría más complejo. Si bien podría solucionarse mostrando los grupos minoritarios en una linea inferior cuando se seleccione la pestaña varios. Esto podría suponer un problema en el caso de que el número de estos grupos sea muy grande, no tanto de no poder mostrarlo, pues las pestañas podrían desplazarse, como de reducirse la claridad, aunque el usuario solo se encontraría con esta situación cuando busque un grupo minoritario, con lo que es de esperar que sea poco habitual, además de esperarse una aceptación por parte del usuario de mayor complejidad, debido a que está buscando temas poco habituales, que sin esta complejidad habría sido imposible incluir.

8. Centroides

En los algoritmos basados en centroides los grupos son generados alrededor de un punto, que es el denominado centroide [5]. El hecho de que un elemento pertenezca a un grupo y no a otro viene determinada en función de la distancia entre el elemento y el centroide de cada uno de esos puntos. Por tanto, hay dos conceptos importantes que difieren entre los diferentes algoritmos basados en centroides, la distancia empleada (la forma de calcular la distancia), y el centroide empleado (el modo en el que se obtiene dicho centroide). Para comprender los algoritmos basados en centroides hemos de conocer en profundidad ambos conceptos, por lo que pasaremos a analizarlos.

8.1. Propiedades de las distancias

En la vida diaria el concepto de distancia es concebido como el intervalo de lugar o tiempo entre dos puntos, elementos, o situaciones. Si bien ésta distancia, o mejor dicho éstas dos distancias, la temporal y la espacial, son aceptadas en el ámbito matemático, el concepto es bastante más amplio. De hecho la distancia espacial empleada comunmente es una distancia en particular que se denomina distancia geométrica. En las matemáticas una distancia, o métrica, es una función que cumple una serie de condiciones particulares que definimos a continuación ([6] Capítulo: Preface):

$$d\left(x,y\right) \geqslant 0 \forall x \wedge y \in X \tag{1}$$

Siendo X el espacio en el cual se aplica la función d.

Es decir, que la distancia entre dos puntos cualesquiera ha de ser un valor real y positivo o cero. El hecho de que el valor tenga que ser real, y no pueda ser imaginario, que no es mostrado explícitamente por la condición, realmente sí que es mostrado de forma implícita, puesto que hemos de tener en cuenta que el concepto de mayor o menor no es aplicable a números del conjunto de los imaginarios, en el cual dos números solo pueden ser iguales o distintos. Lo mismo pasa en los vectores, en los cuales se puede decir que un vector tiene una longitud mayor o menor que otro, así como que tiene algún componente, o todos ellos, mayores que los de otro vector, pero nunca que un vector es a secas mayor o menor que otro.

$$d\left(x,y\right) = d\left(y,x\right) \forall x,y \tag{2}$$

Esta relación es denominada simetría. Indica que la distancia entre dos puntos cualesquiera no depende de cuál de estos puntos es el origen y cuál de ellos es el destino. Ésta condición es muy importante para nuestra aplicación particular de la distancia, puesto que las diferencias, ya sean objetivas o subjetivas, entre dos resultados de una búsqueda no dependen del orden en el que se comparen los citados resultados.

$$d\left(x,y\right) = 0 \Longleftrightarrow x = y \tag{3}$$

Ésta doble implicación nos indica que la posibilidad concebida en la primera ecuación de que la distancia entre dos puntos sea cero solo puede darse en el caso de que se esté tomando entre un punto y sí mismo, siendo estrictamente positiva siempre que ambos puntos diferentes entre sí. Pero no solo eso, además nos indica que siempre que los dos puntos tomados sean el mismo la distancia va a ser cero, es decir que ambos datos son equivalentes, no es que una situación sea un subconjunto de la otra.

$$d\left(x,y\right) \leqslant d\left(x,z\right) + d\left(z,y\right) \tag{4}$$

Esta ecuación es denominada desigualdad triangular, debido a que puede ser concebida como la distancia entre dos aristas de un triangulo por una parte medidas directamente, y por otra tomando como punto intermedio la tercera arista del mismo triángulo.

8.2. Diferentes distancias

Existen muchas distancias distintas entre sí, pero todas ellas cumplen las condiciones anteriormente citadas. Enumeramos las más importantes.

8.2.1. Distancia euclídea

La distancia euclídea es la más empleada en el análisis de vectores. Se basa en la función siguiente:

$$d(x,y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + ... + (x_{n-1} - y_{n-1})^2 + (x_n - y_n)^2}; x \wedge y \in X \tag{5}$$

Siendo x e y vectores de n dimensiones, y siendo x_i el componente i del vector x.

La particularización de esta distancia para vectores de dos componentes, es decir, para $n = 2$ es la denominada distancia geométrica, que es la distancia espacial empleada fuera del ámbito matemático.

Esta distancia puede resultar adecuada en una aplicación de clustering haciendo la consideración de que cada componente del vector es una propiedad, de modo que la distancia total cuantificaría la diferencia en todas las propiedades. La ventaja que aporta esta distancia en el clustering es que permite medir como de diferentes son dos elementos respecto a una propiedad dada. Esto también puede tornarse una desventaja, puesto que hay tanto propiedades cuantificables, como propiedades no cuantificables. Un ejemplo de las primeras sería la altura. Si bien una persona de 1,70m y una persona de 1,71m no miden lo mismo, su distancia es mucho menor que la que tienen ambos con una persona que mida 1,90m. Un ejemplo del caso contrario sería la empresa en la que las susodichas personas trabajan, en la cual solamente podría constatarse si dos personas trabajan en la misma empresa o no.

8.2.2. Distancia discreta

La distancia discreta es la distancia más sencilla concebible respetando las susodichas condiciones. Se basa en la función siguiente:

$$\begin{cases} x = y \Leftrightarrow d(x,y) = 0 \\ x \neq y \Leftrightarrow d\,(x,y) = 1 \end{cases} \tag{6}$$

Es decir, la distancia de un punto a sí mismo es cero, y la distancia de un punto a cualquier otro punto es 1. Si bien esta distancia no nos arroja información sobre "cuánto" de lejos se encuentra un punto de cualquier otro, sí qué tiene una aplicación interesante, nos indica que puntos son distintos entre sí, y cuáles no. Ésta aplicación podría ser interesante en conceptos en los cuales no tiene sentido concebir una distancia mayor o menor en una característica dada de dos elementos distintos, y solo se puede constatar el hecho de que son distintos entre sí. Además tiene la ventaja de que al almacenar solo dos valores posibles, para todo aquello para lo que es aplicable resulta computacionalmente más sencilla y eficiente que otra métrica que conciba más posibilidades.

Un problema añadido de esta métrica, que debe ser tenido en cuenta para nuestro caso en particular, es que solo tiene en cuenta la existencia de diferencia en todo valor del elemento en conjunto, y no en un componente, por lo que podría resultar problemático aún en el caso de que se esté tratando con propiedades de los elementos que cumplen la citada condición de que sus diferencias no sean cuantificables. Esta excesiva sencillez nos negaría la posibilidad de siquiera determinar cuantas propiedades son diferentes entre dos elementos. Esta necesidad nos lleva a la siguiente métrica.

8.2.3. Distancia de Hamming

La distancia de Hamming no se aplica a cualquier espacio, sino que está reducida a los vectores binarios, por lo que en este caso x e y serán siempre vectores binarios. La función que define esta distancia es la siguiente:

$$d\,(x,y) = \sum_{i=1}^{n} |x_i - y_i| \ldots x_i \wedge y_i \in (0,1) \tag{7}$$

Si bien en este caso la distancia entre cada par de componentes vectoriales solo indica si son iguales o diferentes entre sí, puesto que valdrá 1 cuando sean diferentes, y 0 cuando sean iguales, como pasaba en el caso anterior, la diferencia en la distancia total es importante. En este caso sí que estaríamos contando el número de diferencias entre ambos elementos, como deseábamos obtener anteriormente. Eso sí, hemos de hacer una observación importante, en la distancia de Hamming se analizan vectores binarios, no diferentes propiedades de un elemento. De hecho las propiedades de un elemento no tienen porqué ser booleanas, pueden tener más de dos posibilidades, lo que las hace imposibles de cifrar con un solo bit.

Esta distancia es muy empleada en el mundo de las telecomunicaciones, pues permite calcular hasta que punto ha sido modificada una cadena de bits, algo muy útil cuando se trata de ver el efecto de un canal sobre las señales que estamos enviando a través de él.

8.2.4. Distancia de Hamming modificada

Debido a lo que hemos comentado, la distancia de Hamming no puede ser aplicada de modo estricto en nuestro problema, ni siquiera generando un vector binario en el que cada componente fuera una propiedad cifrada. Pero lo que sí que podemos hacer es adaptar esta idea a nuestro intereses. Esto se puede hacer como mostramos a continuación, siendo x e y vectores que representan las propiedades de sendos elementos a ser analizados, a razón de una propiedad por componente:

$$\begin{cases} x_i = y_i \Rightarrow & z_i = 0 \\ x_i \neq y_i \Rightarrow & z_i = 1 \end{cases} \tag{8}$$

$$d(x, y) = \sum_{i=1}^{n} z_i \tag{9}$$

De este modo sí que estaríamos consiguiendo una distancia que fuera el número de propiedades diferentes entre dos palabras. Esta distancia resalta por su sencillez, tanto para bien como para mal. Por una parte esta sencillez tiene ventajas, que enumeramos a continuación:

- Reducido coste computacional.
- Reducidas cantidades de datos a almacenar.

De estas dos ventajas la que realmente tiene importancia es la primera, debido a que nuestra aplicación puede tener que trabajar con una gran cantidad de palabras, por lo que toda reducción de trabajo asociado a una palabra debe ser tenida en cuenta. A pesar de resultar más eficiente computacionalmente que cualquier otra distancia que requiera el cálculo de la distancia entre cada par de palabras, toda aquella función que requiera no emplear tanto cálculo va a ser más eficiente, pero el número de cálculos a realizar, y como optimizarlo lo trataremos más adelante.

Además de las citadas ventajas, hemos de tener en cuenta que este sistema muestra algunos otros problemas que pasaremos a ennumerar:

- Imposibilidad de cuantificar la diferencia en una propiedad dada. Si bien esto puede ser una ventaja cuando las características de la propiedad lo impidan, ha de tenerse en cuenta que se está anulando la posibilidad de hacerlo cuando las características de la propiedad sí que lo permitan.
- Imposibilidad de asignar diferente importancia a cada propiedad. Debemos tener en cuenta que no todas las propiedades a valorar en una distancia

tienen porque tener la misma importancia para nuestros objetivos, pero con este sistema todas ellas son valoradas por igual. Este problema podría solucionarse realizando una modificación que pondere la diferencia de cada una de ellas, pero esto reduciría la ventaja de la eficiencia computacional.

A pesar de tener problemas ésta distancia debe ser tenida en cuenta. Será elegida en caso de no obtenerse otra distancia mejor, y que computacionalmente resulte asequible. Problema, este último, que se podría solucionar de conseguir evitar la necesidad de realizar un cálculo exhaustivo de las distancias.

8.3. Cantidad de distancias a calcular

Un concepto importante de cara al coste computacional es la cantidad de distancias a calcular, ya que una técnica que nos permita tener que calcular muchas menos distancias que otra resultará más eficiente incluso en el caso de que cada una de las distancias a calcular sea algo más compleja. Por esta razón pasaremos a analizar las diferentes posibilidades que pueden darse al respecto.

8.3.1. Método exhaustivo

El método exhaustivo consiste en calcular la distancia entre todo par de palabras. Si bien no es necesario realizarlo una vez que se dispone de los centroides, sí que sirve para obtenerlos. Debemos tener en cuenta que calculando las distancias entre cada par de palabras, cada vez que se añade una palabra nueva ha de calcularse su distancia con respecto a todas las palabras que ya teníamos.

De un modo más matemático, este efecto podría ser mostrado en la sucesión siguiente:

$$x_n = x_{n-1} + (n-1) \tag{10}$$

$$x_2 = 1 \tag{11}$$

Donde $x_i$es el número total de distancias que se han de calcular cuando tenemos i palabras, y n es el número de palabras que tenemos. Empleando un sencillo programa, podemos observar la problemática para algunos números de palabras representativos.

número de palabras	número de distancias a calcular
2	1
5	10
10	45
100	4 950
1 000	499 500
10 000	49 995 000
100 000	704 982 704
1 000 000	1 783 293 664

Cuadro 1: Distancias a calcular de modo exhaustivo con diferentes números de palabras

8.3.2. Distancias elemento-centroide

Una vez obtenidos los diferentes centroides hemos de calcular la distancia entre cada elemento y cada centroide, para determinar a qué grupo pertenece cada elemento. Este número de cálculos es mucho más reducido que el exhaustivo, si bien no sirve como sustituto del anterior, sino como complemento.

El número de cálculos a realizar sigue la función siguiente:

$$x_{n,i} = n \times i \tag{12}$$

Siendo x el número de cálculos a realizar, n el número de elementos, e i el número de centroides.

Como nos muestra la ecuación el aumento del número de cálculos a realizar que implica el aumento unitario en el número de elementos es mucho más reducido en este caso que en el caso exhaustivo, por lo que a efectos computacionales esta fase resulta mucho menos problemática.

Podría considerarse que el hecho de que el número de centroides afecte de forma muy directa a esta ecuación implicaría el que deba ser calculado en función de estos intereses. Pero esto no es así debido a que resulta mucho más importante determinar el número de grupos, que es igual al número de centroides, en función del efecto que esto vaya a tener en la comprensión del usuario, ya que el efecto sobre el tiempo total de procesamiento es bajo, como ya se ha observado, y el efecto de una restricción excesiva en el número de grupos podría dañar seriamente el servicio prestado al usuario.

8.4. Algoritmos que emplean centroides

Los siguientes algoritmos son los más importantes de entre los que emplean centroides.

8.4.1. k-medias

El algoritmo k-medias, más conocido por su nombre en inglés “K-means”, trata de generar k grupos a partir de sendos centroides, para lo que realiza los

pasos siguientes ([7] Capítulo: V.3 Clustering Algorithms):

1. Inicialización:
   Se generan k centroides, que no tienen por qué ser ninguno de los elementos a estudiar. Posteriormente se asigna cada elemento al grupo del centroide con el que tenga menor distancia.

2. Iteración:
   Se recalcula cada centroide como la media de los elementos de ese grupo. Después se reasignan los elementos al grupo del nuevo centroide con el que tengan menor distancia.

3. Finalización:
   En la iteración en la que la reasignación no suponga el cambio de grupo de ningún elemento finaliza el algoritmo.

En cuanto a la generación de los centroides, sobre la que hemos pasado muy de soslayo en el comportamiento del algoritmo, hemos de hacer una aproximación más profunda. El problema básico es la primera generación, puesto que una vez que se alcanza la iteración ya está claro que ha de hacerse la media dentro de cada centroide, lo que se haría con la función siguiente:

$$C_i = \frac{1}{k} \sum_{j=1}^{k} x_{j,i} \tag{13}$$

Siendo C_i el centroide del grupo i, k el número de elementos del grupo i, y $x_{j,i}$ el vector del elemento j del grupo i.

En cuanto a la primera generación, se puede hacer de múltiples formas, si bien es complicado realizar una forma óptima para cada caso, debido a que ha de ser definida a priori para todos los casos de cada problema. Por lo tanto pueden emplearse básicamente dos sistemas. El primero de ellos es el determinista, consistente en crear los centroides como los puntos equidistantes. El segundo de ellos es el puramente aleatorio, en el que se eligen todos los puntos de forma aleatoria, con la única condición de no coincidir varios centroides en el mismo punto (puesto que serían imposibles de separar por la iteración). Una tercera opción, basada en las dos anteriores, es dividir el espacio de estudio en k secciones rectangulares iguales entre sí, y elegir un centroide dentro de cada sección, de forma aleatoria. Reduciendo el determinismo de la primera opción, y también el riesgo de que todos los centroides queden excesivamente cerca, de la segunda opción.

En cuanto a la función empleada para recalcular los centroides que hemos citado, nos aporta una cierta información sobre cómo han de ser las situaciones en las que se emplee este algoritmo. Se está calculando una media, por lo que las propiedades de los elementos han de ser numéricas, ya que resultaría muy complicado hacer la media entre dos nacionalidades, por poner un ejemplo, o entre dos colores, ya que podría hacerse la media entre las amplitudes de onda asociadas, pero eso solo tendría sentido en algunos problemas.

También nos está dando la información de que no basta con que las propiedades sean numéricas, además nos prohíbe que sean booleanas, ya que en un espacio de estudio en el que solo contemos con "1" y "0" no se podría obtener una media entre varios números, puesto que probablemente esta no pertenezca al espacio de estudio.

Además de tener las citadas desventajas de lo reducido de su espacio de aplicación, este sistema tiene otra característica, que podría ser considerada como una ventaja o como una desventaja. Es la obligación, que podríamos llamar posibilidad, de determinar el número de grupos resultantes. Esto puede resultar una ventaja en alguna situación en la que ya sabemos de antemano el número de grupos necesarios. Pero puede resultar una desventaja en la situación en la que haya un número de grupos "natural", que no puedan ser determinados de antemano, por ser diferentes en cada caso. Ésta es precisamente la situación con la que nos encontramos en nuestro problema, debido a que en cada caso hay un número de temas diferentes que pueden ser aludidos por un conjunto de tweets. La obligación de determinar el número de grupos puede darnos tres casos. El primer caso es emplear un número demasiado pequeño de grupos, y podría dar lugar a que elementos de temas minoritarios pero muy alejados de los temas mayoritarios acabarían añadidos a alguno de estos. El segundo caso, es el de emplear un número demasiado grande de grupos, y que acabaría dando lugar a que elementos de un mismo tema acabarían en grupos diferentes, dificultando la comprensión del usuario. El tercer caso sería el de acertar con el número de grupos, pero ha de recordarse que esto solo podría suceder por haber tenido suerte en algún caso en particular.

Una ventaja de este sistema, desde un análisis general, no centrado en nuestro problema, es que este sistema minimiza la distancia interna de cada grupo, por lo que mostraría grupos más reducidos en tamaño. Este hecho nos indica en que situaciones podría ser más útil este algoritmo, nótesen los dos casos siguientes, en los que se puede percibir que en uno de ellos el algoritmo ha funcionado, mientras que en el otro no ha conseguido dar, ni de lejos, con el agrupamiento que los seres humanos habríamos determinado más habitualmente.

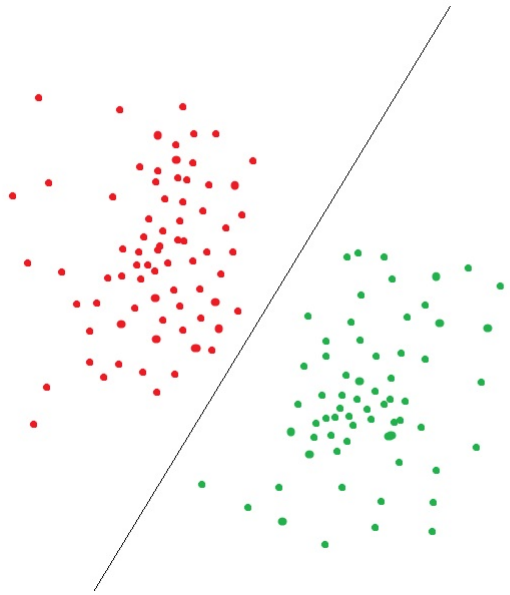

Figura 2: Situación en la que el algoritmo ha coincidido con la percepción humana.

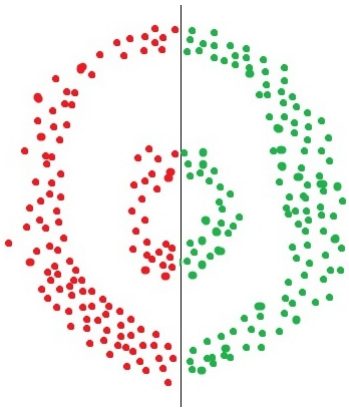

Figura 3: Situación en la que el algoritmo dista mucho de la percepción humana.

Si bien en ambas imágenes resulta sencillo percibir ambos grupos, y además se ha introducido ese mismo número de centroides, resulta evidente que solo en el primer caso el algoritmo ha conseguido determinar esos mismos dos grupos. Esto se debe a que en el segundo caso, no era la distancia en lo que se basaba la discriminación que haría inconscientemente un observador humano, sino en la densidad, ya que se perciben dos círculos concéntricos, siendo un grupo cada

uno de ellos dos. Hay otros algoritmos basados en esta idea que posteriormente analizaremos.

9. Herencia

El concepto de herencia en clustering ya ha sido introducido en su sección correspondiente en el estudio "¿Qué es un grupo?" pero en este caso lo que trataremos son los algoritmos que se basan en este concepto, sus características comunes y diferencias, así como ventajas y desventajas de cada uno de ellos.

Existen dos grandes grupos en los que se dividen los algoritmos de herencia ([2] capítulo 4.1 Introduction). Esta división se realiza en función de si partimos de grupos individuales y vamos juntando hasta obtener un gran grupo, lo que se llamaría aglomeración, o de si partimos de este gran grupo y vamos dividiendo hasta obtener grupos de un solo elemento, lo que se llamaría división. Pasamos a analizar cada caso con la finalidad de descubrir las diferencias entre ambos.

9.1. Aglomeración

Los procedimientos de aglomeración son aquellos que van de grupos más pequeños a grupos más grandes. De una forma más precisa, han de seguir el procedimiento siguiente:

1. Inicialización:
   Cada elemento pertenece a un grupo diferente, al que solo pertenece él.

2. Iteración:
   Los dos grupos con menor distancia entre sí pasan a ser un solo grupo. En caso de ser más de dos con la misma distancia menor, también pasarían a juntarse en el mismo paso.

3. Finalización:
   El algoritmo finaliza cuando hay un solo grupo que contiene a todos los elementos.

Como puede observarse, el resultado final de ambos algoritmos es trivial, y carente de interés, ya que siempre va a ser el mismo. Por tanto, lo interesante no es ese resultado final, sino los pasos que se han seguido para llegar hasta él. Saber en que punto se han unido dos elementos dados, o los grupos que en un punto intermedio del algoritmo se habían formado. Para mostrar esta información, se emplean los llamados dendogramas, que no son otra cosa que arboles en los que se muestra en que punto (o iteración) va convergiendo cada subgrupo. Existen dos tipos importantes de dendogramas, que se diferencian simplemente en que en el primero de ellos todos los elementos son mostrados desde la inicialización, mientras que en el otro cada elemento es mostrado en la iteración en la que pasa a formar parte de un grupo mayor que el formado solamente por él mismo.

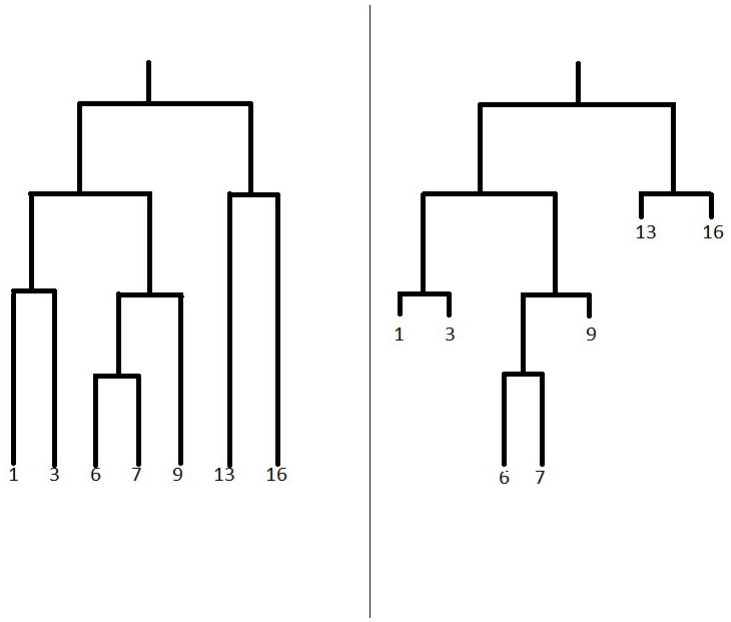

Figura 4: Mismo clustering mostrado por ambos tipos de dendogramas. (Se ha empleado el método del vecino más próximo).

Otro aspecto importante a tratar en los algoritmos aglomerativos es la distancia. En la iteración se cita la distancia como sistema para determinar en qué momento se ha de realizar la unión entre dos grupos dados. En capítulos anteriores se ha tratado el tema de las distancias, analizando varios tipos de distancias posibles entre elementos. Pero en este caso se está tratando de la distancia entre grupos, no entre elementos, pequeño detalle que trae consigo la necesidad de realizar algunas especificaciones. Existen varias posibilidades para calcular la distancia entre grupos ([2] capítulo 4.2 Agglomerative Methods), pasamos a analizarlas:

- Unión simple:
  También conocida como vecino más próximo. Determina la distancia entre dos grupos como la mínima distancia entre elementos de ambos grupos, o de una forma más matemática:

$$d(a,b) = min(d(x_a, x_b)) \tag{14}$$

  Donde a y b son los grupos entre los que se quiere calcular la distancia, y x_i es un elemento del grupo i.
  Produce un efecto denominado encadenamiento, y que consiste en que el ruido entre grupos cercanos entre sí, da lugar a que estos se unan antes de lo que deberían.

- Unión completa:
  También conocido como vecino más lejano. Determina la distancia entre dos grupos como la máxima distancia entre elementos de ambos grupos, lo que matemáticamente se expresaría como sigue:

$$d(a,b) = max(d(x_a, x_b)) \tag{15}$$

  Donde a y b son los grupos entre los que se quiere calcular la distancia, y x_i es un elemento del grupo i.
  Al tenerse en cuenta la máxima distancia entre elementos produce grupos más compactos, además de evitarse el encadenamiento producido por la unión simple.

- Distancia media:
  Este sistema consiste en calcular todas las distancias entre pares de elementos de un grupo cada uno y dar como resultado la media, como mostramos en la función siguiente:

$$d\left(a,b\right) = \frac{1}{k}\sum_{i=1}^{k}\frac{1}{m}\sum_{j=1}^{m} d\left(x_{a,i}, x_{b,j}\right) \tag{16}$$

  Donde a y b son los grupos entre los que se quiere calcular la distancia,k es el número de elementos del grupo a, m es el número de elementos del grupo b, $x_{a,i}$ es el elemento i del grupo a y $x_{b,j}$es el elemento j del grupo b.
  Este sistema podría ser considerado un punto intermedio entre los dos anteriores, porque se fija en todos los elementos, no solo en los más extremos. Podría considerarse que tiene un coste computacional claramente superior a los anteriores, por requerir el cálculo de todas las distancias entre los dos grupos, pero realmente no es así, puesto que el algoritmo también tiene que hacer todas esas operaciones para descubrir en los anteriores cuál es la menor y la mayor de ellas, respectivamente. La única diferencia es que en los anteriores desecha todos los cálculos menos uno, lo que no afecta al hecho de tener que haberlos realizado.

- Distancia entre centroides:
  Este método emplea un sistema conceptualmente parecido al anterior, pero tratando de reducir el coste computacional de éste. Se basa en obtener primeramente un centroide de cada grupo, y posteriormente tomar la distancia entre centroides como distancia entre grupos. Se apoya en las dos funciones siguientes:

$$C_i = \frac{1}{k}\sum_{j=1}^{k} x_{j,i} \tag{17}$$

  Siendo C_i el centroide del grupo i, k el número de elementos del grupo i,

y $x_{j,i}$ el vector del elemento j del grupo i.

$$d\left(a,b\right) = d\left(C_a, C_b\right) \tag{18}$$

Si bien reduce notablemente el número de cálculos a realizar en un paso dado tiene un problema, y es que el centroide ha de ser recalculado cada vez que se modifica el grupo, mientras que los pares de distancias son los mismos durante todas las iteraciones, por lo que solo han de ser calculados la primera vez. Una solución para reducir el coste computacional de este método es calcular el centroide resultante de una unión como la media ponderada de los centroides de los grupos de los que se partía al comienzo de la iteración, siendo el peso de cada centroide el número de elementos de su grupo. La igualdad entre ambos métodos se demuestra a continuación:

$$C_i = \frac{1}{k}\sum_{j=1}^{k} x_{j,i} = \frac{1}{k}\left(\left(\sum_{j=1}^{m} x_{i,j}\right) + \left(\sum_{j=m}^{k} x_{i,j}\right)\right) =$$

$$= \frac{1}{m+(k-m)}\left(m\left(\frac{1}{m}\sum_{j=1}^{m} x_{i,j}\right) + (k-m)\left(\frac{1}{k-m}\sum_{j=m}^{k} x_{i,j}\right)\right) =$$

$$= \frac{(m \times C_a + (k-m) \times C_b)}{m+(k-m)} \ldots i = a \cup b \wedge m < k \tag{19}$$

Siendo los m primeros elementos del grupo i los pertenecientes a a y los restantes los pertenecientes a b.

9.2. División

Los algoritmos de división son los opuestos a los de aglomeración, dentro del clustering de herencia. Si bien a grandes rasgos podemos considerarlos, como ya hemos dicho, opuestos a los de aglomeración, tienen algunas características particulares, que les separan de ser perfectamente simétricos.

Al igual que en el caso de los algoritmos de aglomeración, la gran mayoría de la información no es mostrada por el resultado final del algoritmo, aunque en este caso si que puedan aportar alguna información en algunas ocasiones

Existen dos tipos básicos de algoritmos de división ([2] Capítulo: 4.3 Divisive Methods), que son los monotemáticos, más conocidos por su nombre en inglés que es "monothetic", y los politemáticos, a su vez más conocidos como "polithetic". Pasamos a analizarlos.

9.2.1. Monotemáticos

Los algoritmos monotemáticos son aquellos que realizan la división en función de una sola de las propiedades de los elementos en cada caso, o también podríamos decir, en función de uno solo de los componentes del vector de cada elemento. El funcionamiento que siguen es el siguiente:

1. Inicialización:
   Se parte de un gran grupo que contiene a todos los elementos a analizar.

2. Iteración:
   Se realiza una división de cada uno de los grupos existentes en función de una propiedad no empleada anteriormente.

3. Finalización:
   En el momento en que o bien todas las propiedades han sido empleadas, o bien no existen grupos con más de un elemento, el algoritmo finaliza.

Un punto importante a determinar en estos algoritmos es la propiedad a analizar en cada caso, puesto que esta decisión no es nada trivial con respecto al resultado ofrecido por el algoritmo. Una opción puede ser la selección aleatoria de ésta, pero al tratarse de algo tan importante podría no ser la mejor. Otra opción, que no siempre puede ser realizada, es la elección de esta propiedad en función de un orden de propiedades de mayor a menor importancia. Esta última opción, mucho más lógica que la primera, hay casos en los que no puede ser realizada debido a que en algunos casos resulta carente de sentido dar mayor importancia a unas propiedades que a otras. Una tercera opción, más objetiva que la segunda, es la elección de la propiedad a tener en cuenta en cada caso en función de un criterio general, que no requiera del conocimiento de cada caso en particular. Un criterio a tener en cuenta es el mostrado por Lance y Williams en 1968 ([2] Capítulo: 4.3 Divisive Methods). Este algoritmo trata de maximizar la información aportada por cada iteración, de modo que se minimice el número de iteraciones a realizar, para lo que emplea la función siguiente:

$$C = pn \log n - \sum_{k=1}^{p} \left[f_k \log f_k - (n - f_k) \log (n - f_k) \right] \tag{20}$$

Se elegirá la división que maximice C.
Donde C es la información aportada por la división en un grupo dado, p es el número de propiedades que puede tener un elemento, n es el número de elementos de un grupo dado, y f_k es el número de elementos de ese grupo que tienen un "1" (ver desventajas a continuación) en la propiedad k.

Estos métodos tienen unas ventajas y unas desventajas muy marcadas que deben ser valoradas antes de elegirlos o desecharlos. La principal ventaja que tienen es que a partir de un punto en el árbol se puede garantizar que todos los elementos pertenecientes a un grupo dado, o a un conjunto de grupos dados, tienen un cierto valor en una cierta propiedad.

Otra ventaja a tener en cuenta es la reducida complejidad computacional que requieren sus operaciones, puesto que no requieren de realizar ningún cálculo de distancia, como sí que ha de ser realizado en otros casos.

La principal desventaja que tienen éstos algoritmos es su reducido ámbito de aplicación. Este problema se debe a que están concebidos para elementos con propiedades booleanas (de ahí que en la susodicha función se requiera el número de elementos que tienen un 1). De este modo se produce un árbol binario, en el que en cada iteración cada grupo se divide en dos, siempre y cuando existan elementos con ambas opciones en cada uno de los grupos de origen de la iteración.

La segunda desventaja a la que hemos de prestar atención es la excesiva variación que puede sufrir un algoritmo en función del orden en el que se tomen las propiedades. Este hecho provoca dos efectos, el primero de ellos la gran diferencia que se puede encontrar entre arboles que provengan del mismo conjunto de elementos, lo que provoca cierta desconfianza sobre la información mostrada. El segundo efecto es el hecho de que dos elementos con propiedades muy similares puedan separarse en una iteración mucho anterior que dos elementos con propiedades muy diferentes, por darse las pocas diferencias entre los primeros y las pocas coincidencias entre los segundos en las propiedades analizadas en las primeras iteraciones.

En cuanto al reducido espacio de aplicación de éstos algoritmos, se puede tratar de realizar algunas soluciónes parciales. La primera de ellas, y la más sencilla, es la de aceptar propiedades con un número finito (y mejor si es reducido) de opciones. Esto podría hacerse simplemente con la aceptación de que el árbol deje de ser binario, pasando a crearse n subgrupos como máximo a partir de cada grupo de partida en cada iteración, siendo éste el número de posibilidades de la propiedad analizada en cada caso, pudiendo n variar en cada iteración. En esta solución ha de tratarse que el número de posibilidades de cada propiedad sea claramente inferior al número de elementos a analizar, puesto que de no ser así podría darse el caso de árboles muy cortos y anchos en los que en excesivamente pocas iteraciones se han posicionado la mayoría de los elementos, si no todos, en grupos individuales, de modo que muchas propiedades han quedado por ser analizadas.

Una segunda solución, más compleja que la anterior, y que permitiría el uso de estos algoritmos con propiedades con infinitas opciones, como por ejemplo una que emplee un intervalo de la recta real, es la de crear fronteras. Esto es, determinar, por diseño, unos valores para cada propiedad, respecto a los cuales se dividirían los elementos en función de ser mayores o menores (situando el ser iguales en uno de los dos grupos). Podrían crearse tanto una sola frontera, con lo que se conseguiría un árbol binario, como varias, empleándose también la primera solución. De quererse realizar un algoritmo que permita ser usado en situaciones con valores muy diferentes podrían emplearse fronteras que no hayan sido diseñadas en función de un valor previamente impuesto, sino que usen proporciones, de modo que fuercen a que el x % de los valores más altos pertenezca a un grupo, el x % al siguiente, y así hasta completarse la división.

9.2.2. Politemáticos

Los algoritmos politemáticos tienen un funcionamiento que sí que resulta más simétrico respecto a los de aglomeración. Estos algoritmos no van analizando las propiedades de una en una, sino que estudian todas ellas en conjunto en todos los casos. Para ello emplean el funcionamiento siguiente:

1. Inicialización:
   Todos los elementos pertenecen un mismo grupo.

2. Iteración:
   El grupo cuyos elementos tengan la mayor distancia interna se divide en dos. En caso de haber varios grupos con la misma mayor distancia interna se dividirían todos ellos en la misma iteración.

3. Finalización:
   El algoritmo finaliza en el momento en el que solo existen grupos con un solo elemento.

Una vez más, e igual que nos sucedió en el caso de los algoritmos aglomerativos, volvemos a encontrarnos con el problema de aplicar las distancias a un grupo. En este caso con la problemática añadida de que hay que considerar las distancias de dos grupos que no existen todavía, y que de hecho han de ser deducidos por el propio algoritmo.

Una posible solución a esta problemática sería la de calcular todos los posibles pares de grupos resultantes, calculando en cada caso la distancia entre ellos, con alguna de las opciones vistas en los algoritmos aglomerativos, y tomar el resultado en el que tengan una mayor distancia intergrupal. El principal problema de ésta solución es el enorme coste computacional que conlleva, ya que para una sola iteración han de calcularse múltiples subgrupos de cada uno de los grupos, además de la distancia entre ellos. Debido a esta problemática conviene acudir a otras soluciónes más sencillas.

Una solución que requiere de mucho menor coste computacional se apoya en la idea del vecino más lejano estudiada anteriormente. Se basa en tomar los dos elementos más alejados de cada grupo, y crear los subgrupos a partir de éstos, asociando a cada elemento los más cercanos, ya sea tomando los más cercanos al elemento original en cada caso, o los más cercanos a alguno de los elementos que se han ido añadiendo a cada grupo. Si bien esta solución puede no dar un resultado óptimo como sí que daría la anterior, el enorme coste computacional que requiere la otra la convierte en una buena opción.

Una desventaja que tienen éstos algoritmos es, al igual que en el caso de los monotemáticos, su espacio de aplicación. En este caso el espacio natural de aplicación de estos sistemas son los elementos con propiedades numeradas, que permitan calcular distancias. Si bien se han estudiado distancias que son perfectamente aplicables a vectores booleanos, se podría correr el riesgo de que se dé la misma distancia máxima entre varios pares de elementos dentro del mismo grupo, siempre que el número de propiedades no sea muy alto, lo que

nos pondría en la tesitura de tener que elegir entre varias opciones que llevarían a arboles distintos entre sí.

El hecho de que ambas opciones, monotemáticos y politemáticos, tengan espacios acotados de aplicación, pero que estos espacios requieran características opuestas de las propiedades podría hacernos pensar que estos son complementarios, pero existe un conjunto a tener en cuenta de elementos que no son aceptados (sin el empleo de las citadas soluciónes) por ninguno de ellos. Éste es el conjunto de los elementos que tienen propiedades de múltiples tipos, unas booleanas, otras de un conjunto mayor pero acotado de opciones, otras asociadas a un valor numérico cualquiera... Puesto que se requiere que todas las propiedades cumplan las características de cada algoritmo, y no solo alguna de ellas.

10. Distribución

En muchos ámbitos en los que se emplea el clustering, de lo que estamos realmente tratando es de descubrir varias poblaciones que se nos presentan mezcladas. Cada una de éstas poblaciones ha sido generada, ya sea de forma natural o artificial, con una distribución aleatoria. Por tanto, una buena forma de separar éstas poblaciones sería descubrir cuáles son esas distribuciones. Con este espíritu apareció un modo de clustering denominado de distribución.

Este clustering trata precisamente de descubrir por diferentes técnicas cuáles son esas distribuciones que han dado lugar a los elementos de estudio, y además, descubrir a cuál de esas distribuciones pertenece cada elemento. Hemos de entender que éstas distribuciones van a tener algo distinto entre sí, cuando hablamos de distribuciones del mismo conjunto de estudio, puesto que de tener todas las propiedades iguales resultarían imposibles de distinguir, y tendríamos que considerarlas como una sola distribución que contendría a los elementos de ambas, aunque realmente tuvieran dos orígenes, y no uno. Las diferencias entre éstas distribuciones pueden ser, por una parte, el tipo de distribución, entiéndase normal, de Bernouilli, de Raylegh... O, por otra parte diferentes valores de las mismas distribuciones, es decir, diferente media, diferente varianza...

El caso más complejo de los dos es el primero, aquel en el que se emplean varios tipos de distribuciones en el mismo espacio de estudio. Este hecho aporta gran complejidad, sobre todo porque hay que deducir que distribuciones se están empleando, y además qué valores tiene cada una de ellas.

En el segundo caso se está asumiendo que todas las distribuciones de la muestra son de la misma forma, con lo que solo habría que obtener los valores de cada una de ellas (así como el número de distribuciones, en el caso general). El problema de este caso es de donde obtener el dato del tipo de distribuciones que componen el espacio muestral. Este dato puede ser obtenido del conocimiento previo del tipo de problema, lo que ya nos llevaría a cada caso en particular, o de una simplificación, que también requeriría de datos que nos hagan confiar en que no se aporta un error excesivo. Datos que pueden ser empíricos, es decir, que una técnica podría ser la de realizar el estimador suponiendo una distribución sin tener ninguna garantía, y posteriormente comprobar que funciona.

En primer lugar estudiaremos las distribuciones que se emplean más habitualmente, con la finalidad de que cuando se estudien los métodos empleados en el clustering de distribución se pueda hacer un uso técnico de ellas.

10.1. Distribución normal

La distribución normal es la más empleada en estadística, debido a que es la que modela muchos de los procesos de la naturaleza así como de muchos procesos que involucran al ser humano. Pero no solo, sino que también es muy utilizada porque, además, es la que emplea el teorema central del límite ([8] Capítulo: 3. The Central Limit Theorem), tan empleado en estadística, y que pasamos a describir a continuación:

> La suma de variables aleatorias independientes entre sí e idénticamente distribuidas (v.a. i.i.d.), tiende a una distribución normal cuando el número de muestras es lo suficientemente grande.

esto implica que en muchos procesos que ni siquiera siguen la distribución normal, ésta pasa a ser la empleada por el simple hecho de que estamos trabajando con la suma de muestras, y no con las muestras en sí. Ventaja que se acrecienta en el caso de que no conozcamos la distribución que siguen las muestras, pues en el momento en el que podamos garantizar que todas ellas siguen la misma distribución, y que además éstas muestras son independientes entre sí, ya tendremos una distribución con la que trabajar sobre su suma.

Pero centrémonos en la distribución en sí. La distribución normal también es denominada gausiana, y sigue la siguiente función de densidad de probabilidad, que mostraremos primero en una dimensión, y luego en n dimensiones:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{21}$$

$$f(X) = \frac{1}{(2\pi)^{\frac{N}{2}}\sqrt{|R|}} \exp\left(-\frac{1}{2}\right)(X-M)^T R^{-1}(X-R) \tag{22}$$

Siendo x (Minúscula) la variable unidimensional de la que estamos mostrando su distribución, σ^2 su varianza, y μ su media en cuanto a la primera función.

Y siendo X (Mayúscula) el vector de N dimensiones del que estamos mostrando la distribución, R la matriz de covarianzas y m la matriz de esperanzas, en cuanto a la segunda función.

Como puede observarse la complejidad de la función de densidad aumenta notablemente según va aumentando el número de dimensiones del espacio de estudio, por lo que resultaría de gran utilidad conseguir reducir este número siempre que sea posible, y que además no suponga un gran error añadido.

Pero el gran problema no es el número de dimensiones de la función de densidad, sino su integral, la denominada función de distribución, puesto que dista mucho de ser trivial, razón por la cual se emplean tablas que ya contienen los valores de esta integral, denominada función Q. Ésta función emplea una tabla para la distribución normal de media 0 y varianza 1 que indica el área entre el origen y el punto sobre el que se aplique la función, pero puede ser empleada para cualquier distribución normal unidimensional de cualquier media y varianza, en la cual se estaría mostrando el área contenida bajo la función entre la media y el citado punto, a partir de la normalización siguiente:

$$Q\left(\frac{|x-\mu|}{\sigma}\right) \tag{23}$$

Donde x es el punto en el que se desea calcular la integral, μ es la media de la distribución, y σ es su desviación típica, o lo que es lo mismo, la raíz de la varianza.

10.2. Distribución t de Student

La distribución de Student, más conocida como "t", toma su nombre del seudónimo Student empleado por William Sealy Gosset, químico que trabajaba en el departamento de control de calidad de la fábrica de cerveza de Guinness en Irlanda ([8] capítulo 6. The t-test). Esta distribución se emplea cuando hay un número de muestras reducido para emplear el teorema central del límite, pero se es consciente de que las muestras siguen una distribución normal.

Si bien esta distribución es estéticamente muy parecida a la distribución normal, además de compartir con ésta el ser simétrica con respecto a su media, su obtención es completamente diferente. Ésta se basa en los grados de libertad de una muestra, que no son otra cosa que el número de variables que pueden cambiar de valor, también conocidos como tamaño de la muestra.

La distribución sigue la función siguiente:

$$t = \frac{\overline{x} - \mu}{\sqrt{\frac{s^2}{n}}} \tag{24}$$

Donde t es la probabilidad de obtener una media dada, $\bar{x}$ es la media de cada muestra, μ es la media de la población, s^2 es la varianza muestral de la población, y n es el número de elementos de la muestra.

La aplicación más habitual en la que se emplea esta distribución es el test de Student, o t-test. Éste se emplea para tratar de demostrar si es mentira que una distribución tenga una esperanza dada, o bien si es mentira que su esperanza sea menor o igual a un valor. Ha de tenerse en cuenta que se trata de demostrar que es mentira, no que sea verdad, puesto que de determinarse que no pueda decirse que sea mentira, no se estará determinando que sea verdad. Un ejemplo muy claro es el caso en el que se afirme, a sabiendas de estar mintiendo, que la esperanza es un valor muy cercano al real (muy cercano en términos de la varianza de la distribución). En ese caso es altamente probable que se determine que no se puede decir que la afirmación sea mentira, pero tampoco se podría decir que es verdad, de hecho en el ejemplo sabemos que no lo es. Otro punto importante que debe ser tenido en cuenta es que no se determina la falsedad de las afirmaciones de forma categórica, sino con una cierta probabilidad, es decir, dando lugar a resultados del tipo "con un 95 % de confianza (probabilidad) se puede afirmar que la afirmación de que la media es no es menor que 1 es falsa". Cuanto mayor grado de confianza menor probabilidad de que estemos afirmando que es falso algo que no lo es, pero mayor probabilidad de que estemos diciendo que no podemos decir que es falso algo que sí lo es. Razón por la cual no se emplea la probabilidad del 100 %, que nos llevaría a no poder rechazar ninguna afirmación, por alejada de la realidad que ésta nos parezca.

En este test calculamos el intervalo de confianza en el que puede estar la muestra con una probabilidad dada, para lo que empleamos las dos ecuaciones siguientes:

$$CI_{1-\alpha} = \bar{x} \pm \left(t_{\frac{\alpha}{2},n-1}\right)\sqrt{\frac{s^2}{n}} \tag{25}$$

Donde,$CI_{1-\alpha}$ es el intervalo de confianza$1-\alpha$, $\bar{x}$ es la media muestral, que será el centro del intervalo, $t_{\frac{\alpha}{2},n-1}$ es una distribución t con valores $\frac{\alpha}{2}$ y $n-1$ donde n es el número de muestras y $1-\alpha$ la confianza ($\alpha \in [0,1]$) y s^2 la varianza muestral.

La t que citamos puede ser obtenida de una tabla, en función de los dos valores que contiene.

10.3. Estimador de máxima probabilidad

Este estimador, más conocido como ML por sus siglas en inglés, Maximum Likehood, se basa en obtener a posteriori la distribución de la que hay la máxima probabilidad de que provenga un elemento dado, es decir, los parámetros de la distribución, no así el "formato" de ésta. Para ello empleamos la función que muestra la probabilidad a posteriori de que las observaciones provengan de una distribución dada, que es la que sigue:

$$L\left(\theta \mid x_1, ..., x_n\right) = \prod_{i=1}^{n} f_\theta\left(x_i\right) \tag{26}$$

Es decir, que la probabilidad de que un conjunto de n elementos x provenga de una distribución θ es igual al producto de las distribuciones de probabilidad de θ particularizadas en cada x_i .

Puesto que esta ecuación es positiva (debido a que las probabilidades negativas no existen), sus máximos coincidirán con los de su logaritmo. Este hecho, junto a la presencia, computacionalmente problemática, de un productorio, nos lleva a emplear su logaritmo, como sigue:

$$\ell\left(\theta \mid x_1, ..., x_n\right) = \ln L\left(\theta \mid x_1, ..., x_n\right) = \sum_{i=1}^{n} \ln f_\theta\left(x_i\right) \tag{27}$$

Como podemos observar el productorio ha desaparecido en favor de un sumatorio de logaritmos neperianos, reduciéndose la dificultad. Como hemos comentado, el estimador se basa en la búsqueda de la máxima probabilidad, por lo que se expresa como sigue:

$$\theta_{ML} = \arg \max_{\theta \epsilon \Theta} \ell\left(\theta \mid x_1, ..., x_n\right) \tag{28}$$

Siendo la θ_{ML} obtenida la distribución a la que los elementos $x_1, ..., x_n$ pertenecen.

Debido al hecho de que con este sistema se pueden calcular los parámetros de una función, pero no su aspecto se considerará que las distribuciones son normales excepto en dos casos. El primero de ellos es que se cuente con un número bajo de elementos por cluster, considerando como bajo menor que 30, y el segundo que las funciones de distribución sean conocidas a priori, situación en la cual se emplearán estas distribuciones. En el caso de tener un número bajo de elementos se emplearán las distribuciones de Student, que ya han sido citadas anteriormente.

11. Densidad

El último enfoque empleado por los algoritmos de clustering que analizamos es, en cierto modo, el más natural. Esto es, el que resulta más aproximado al clustering inconsciente que realiza un ser humano con solo observar un conjunto de elementos. Sin embargo no hemos de dejarnos llevar a engaño, esto no implica en ningún modo que estos algoritmos sean intrínsecamente mejores que los analizados anteriormente. Una vez más, depende del campo de trabajo, por lo cual hemos de analizarlos, con la finalidad de conocer sus ventajas y desventajas.

Un algoritmo basado en densidad es aquél que considera como elementos del mismo grupo a aquéllos que están más cercanos a un gran número de elementos. Por tanto, esta consideración depende de conceptos subjetivos, como son la cantidad de elementos mínima para considerar que un grupo de elementos es "un gran número" y también la distancia máxima que puede ser considerada entre dos puntos para aceptar que estos están "cercanos entre sí".

De la definición que hemos citado se deducen la principal ventaja de estos grupos, que es la ya citada. Todo elemento que pertenezca a un grupo va a estar enlazado a todos los demás en una especie de malla formada por elementos del propio grupo, habitualmente múltiples elementos. Ésta es la razón por la que estos grupos coinciden con la perspectiva humana, ya que ésta es la forma más sencilla en la que percibimos grupos de forma inconsciente. En la imagen se puede observar un conjunto de elementos que se trató de analizar con el uso de centroides, pero quedó lejos de mostrar los grupos "naturales" que existían, puede observarse que en este caso sí que coincide el resultado con el esperable:

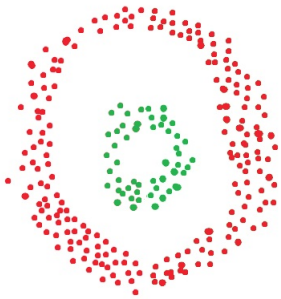

Figura 5: Grupos concéntricos analizados por un método basado en densidad.

Otra ventaja que se deduce de la citada definición es la reducción notable de un efecto perjudicial que existía en los grupos de herencia basados en unión.

Esto es, la unión de grupos diferentes por el efecto del ruido, o por la excesiva cercanía entre elementos de los extremos de ambos grupos gracias a la exigencia de que un elemento tenga a unos cuantos a su alrededor para incluir a todos ellos dentro de un grupo reduce este efecto, aunque no lo elimina. Podría darse que un elemento de un extremo tenga a muchos a su alrededor, siendo alguno de ellos del otro grupo, pero es más improbable, puesto que los elementos de los extremos habitualmente se encuentran rodeados de áreas vacías, que reducen notablemente el número de elementos que tienen a su alrededor.

Una característica, que puede ser considerada como un problema, que tienen estos algoritmos es que al exigir que un elemento tenga muchos en su entorno o bien esté en el entorno de un elemento que los tenga, provoca que existan elementos que no pertenezcan a ningún grupo. Como comentamos en el análisis de las posibles propiedades de un grupo, ésta es una posibilidad perfectamente válida, pero puede resultar problemática para algunas aplicaciones, por lo que siempre debe ser tenida en consideración.

11.1. Algoritmos que emplean densidades

El algoritmo que más se ajusta a la definición que hemos realizado, es el denominado Mode analysis ([2] Capítulo: Miscellaneous clustering methods), que describimos a continuación:

1. Inicialización:
   Se definen tres variables, la primera de ellas n será el número mínimo de elementos que han de estar cerca de un elemento para que este sea considerado denso, la segunda k será empleada en el cálculo de distancias mínimas como se definirá posteriormente, y la tercera d será la distancia mínima entre elementos para considera que estos se encuentran cercanos entre sí. Se considera otra distancia mínima l, siendo ésta la media entre las $2k$ mínimas distancias entre elementos.

2. Iteración:
   Se buscan todos los elementos que tienen en un entorno de distancia d un mínimo de n elementos. Los elementos seleccionados pasan a ser considerados elementos densos.
   - Si dos o más elementos densos se encuentran a una distancia menor o igual a l entre sí estos elementos pasan a formar un grupo común.
   - A su vez, si los elementos que se encuentran en el entorno de un elemento denso se encuentran a la vez en el entorno de otro u otros elementos densos, todos ellos pasarán a formar parte del mísmo grupo.
   - En caso de no cumplirse ninguna de las situaciones anteriores, el elemento denso y todos los de su entorno pasarán a formar parte de un nuevo grupo.
   Por último, el valor de d es aumentado en una cantidad constante en cada iteración.

3. Finalización:
   Cuando una iteración no de lugar a la aparición de ningún elemento denso

nuevo finalizará el algoritmo.

Una vez hemos obtenido el resultado final del algoritmo hemos de analizar sus resultados. De haberse obtenido un solo grupo, se puede considerar que el espacio de estudio no tiene ninguna división natural basada en densidad. En cambio, en el caso de que sí existan múltiples grupos, como mínimo dos, puede considerarse que sí que existe una división natural de los resultados.

Un posible problema de este grupo es que tiene dificultad para generar a la vez grupos pequeños de alta densidad y grupos grandes de baja densidad. Puesto que los primeros son generados rápidamente, y los segundos requieren de múltiples iteraciones, corriéndose el riesgo de que los pequeños pasen a formar parte de los grandes debido a que el aumento de la variable d provoque como resultado que la distancia entre grupos, considerable desde la perspectiva del grupo pequeño, pase a no resultar suficiente.

Este problema ha de ser muy tenido en cuenta en nuestro caso, puesto que las temáticas pueden estar muy cercanas pero perfectamente diferenciadas en situaciones en las que se dan muchos tweets sobre ellas, como puede ser el caso de un tema de actualidad, pero muy separadas en otras temáticas en las que se den muchos menos tweets al respecto. Muchas veces, uno de los problemas con los que nos encontramos, es que queremos separar una temática que comparte términos con otra sobre la que hay muchos más tweets, por lo que hemos de confiar en el algoritmo generado para poder acceder a la temática minoritaria, si volvemos a caer empleando el algoritmo en el mismo problema que teníamos inicialmente no nos va a resultar en absoluto de ayuda.

Parte III
Estrategias en el desarrollo de la solución

Se plantea el desarrollo práctico de una solución que emplee los algoritmos estudiados, tratando de maximizar el resultado. La solución ha de ser capaz de gestionar un conjunto de tweets que se le introduzcan y separarlos por temática en el número de grupos que así se le indique.

Con la finalidad de poder realizar una comparación se emplean dos algoritmos muy diferentes entre sí, de modo que se puedan obtener dos enfoques diferentes de la solución. Éstos son el algoritmo de k-medias y el algoritmo de herencia.

El primero, el algoritmo de k-medias es un algoritmo que se basa en la generación de tantos puntos, denominados centroides, como grupos queramos tener, y la asociación de cada elemento al grupo compuesto por su centroide más cercano, posteriormente el centroide se recalcula como el punto medio de todos los componentes de su grupo, y se vuelve a asignar cada elemento al grupo de su centroide más cercano. Repitiéndose el bucle hasta que se dé una iteración en la que no se produzca ningún cambio, momento en el que se da por finalizado el algoritmo, tomándose como resultado final la situación en ese momento. Este algoritmo será descrito con mayor precisión en la sección Algoritmo elegido.

El segundo, el algoritmo de herencia aglomerativa es un algoritmo que parte de grupos iniciales compuestos por un solo elemento cada uno de ellos, y va juntando los grupos que se encuentran a la mínima distancia entre sí de entre todos los grupos. El bucle se repite, en el algoritmo original hasta obtener un solo grupo, pero en el nuestro hasta obtener el número de grupos deseado, que, recordemos, se puede elegir.

Este algoritmo será descrito con mayor precisión en la sección Otra posible solución.

A continuación se procederá a describir cada una de las soluciones así como sus resultados.

12. Descripción técnica

Para la implementación se ha elegido el lenguaje Java de programación, por resultar ya conocido y permitir más dinamismo que la implementación dentro del programa Matlab, que también estuvo sobre la mesa. El programa se estructura en cuatro clases, tratándose de conseguir una estructuración coherente que permita la fácil comprensión del algoritmo, lo que a su vez ha de dar lugar a una mejor detección de los fallos. Así como un empleo coherente del concepto de objeto en la programación orientada a objetos a la que pertenece el lenguaje empleado. La división en clases permite además la implementación de múltiples algoritmos con mayor sencillez. Este hecho se debe a que con la estructuración

que tiene el programa, las variaciones producidas por la elección de un algoritmo u otro se ciñen a una de las clases, por lo que añadiendo a esa clase los métodos necesarios para implementar otro algoritmo,o bien creando una nueva clase que sustituya a la que implementa los algoritmos actuales, se puede volver a usar completamente todo el resto de clases.

El programa responde a la siguiente estructura:

Figura 6: Diagrama de bloques programa completo

Pasaremos a mostrar el diagrama de cada clase, describiendo posteriormente los métodos más relevantes:

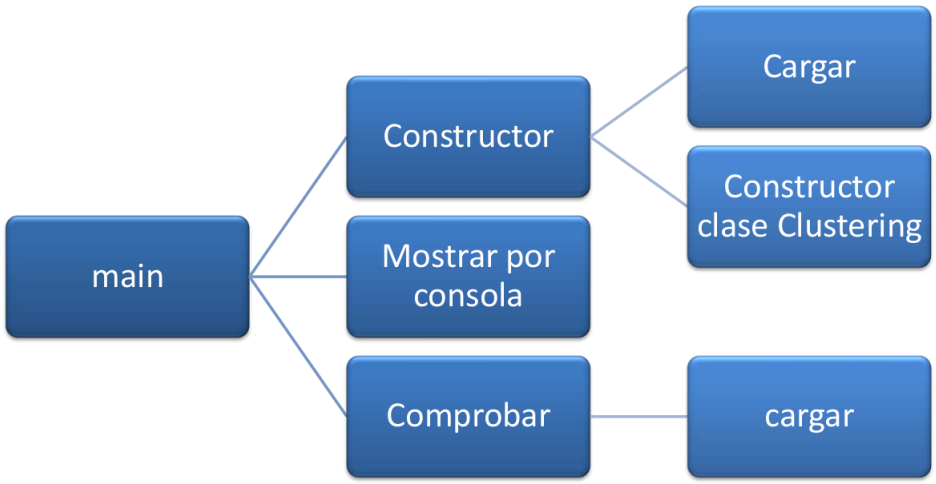

Figura 7: Diagrama de bloques clase Principal

Principal Como bien indica su nombre, esta clase es la que contiene el método main, que es el primero que se inicia cuando arranca el programa. Además, contiene un constructor, un método que llama al cargador de los datos, un método que muestra por pantalla la selección de clustering una vez realizada y otro método que es el que permite comprobar los resultados. También se ha de tener en cuenta que esta clase es en la que se determinan los parámetros que luego se transmitirán a otras clases. Estos parámetros son los siguientes:

- Número de grupos a generar al final del resultado: Se permite cualquier valor natural (Sin incluir el cero), aunque valores altos implican ejecuciones más lentas.
- Ficheros que serán cargados: Se permite cargar cualquier número de ficheros en formato JSON que contengan los identificadores "id" y "text".
- Idioma en el que se encuentran los documentos: Solo se implementa el idioma español, aunque con la sola generación de un fichero en formato TXT con las palabras de cada idioma que no son necesarias para la división temática, como puedan ser preposiciones, artículos o pronombres, se conseguiría una implementación completa para trabajar con documentos en otro idioma. Resulta de ayuda, en el caso de haber errores habituales de esas palabras incluirlos también como se ha hecho en el caso del español, puesto que mejorarían el resultado del clustering.
- Algoritmo de clustering que se empleará: Se implementan los algoritmos de k-medias y de herencia aglomerativa por método de centroides.
- Método para seleccionar las semillas: El algoritmo de k-medias requiere unos valores de centroides iniciales, a partir de los que comenzar el bucle. Se permite en el programa la ya citada elección entre el método aleatorio y el método centrado, y ésta ha de realizarse también en la clase Principal.

La clase principal llama a su vez a dos clases, la primera de ellas es la clase Cargador, que le devuelve un objeto Cargador el cual contiene un array de Strings.

Una vez obtenido el array de Strings, la clase Principal llama a la clase Clustering introduciéndole todas las susodichas elecciones así como el array de Strings, con la finalidad de que ésta le devuelva un objeto Clustering que contiene la selección de los Strings en grupos. Una vez obtenida la selección se la pasa como parámetro a un método cuya finalidad es mostrar el resultado por pantalla, y por último la vuelve a pasar como parámetro, pero en este caso a un método cuya finalidad es comprobar la calidad de los resultados, mostrando las coincidencias entre cada uno de los grupos de entrada y cada uno de los grupos de salida, entiéndase, los grupos obtenidos de la citada selección contenida en el objeto Clustering.

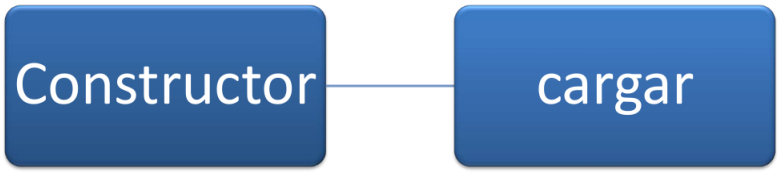

Figura 8: Diagrama de bloques clase Cargador

Cargador La finalidad de esta clase es obtener un array de Strings a partir de un fichero en formato JSON, formato empleado debido al hecho de que en el API que se está desarrollando para Twitter es el único formato soportado, sumado al hecho de que ya está soportado por el API anterior, anque en este caso junto a XML [9]. Para lograr tal finalidad contiene un constructor, que requiere como argumento el nombre del fichero que será cargado, así como un único método que también requiere el mismo nombre de archivo como parámetro. Este método carga el archivo citado, gestiona el archivo JSON obteniendo los valores que responden a los identificadores "id" y "text" y junta cada dupla de ellos en un único String. Una vez obtenidos todos ellos, los aúna a su vez en un array de Strings, que es incluido como propiedad del objeto Cargador que se devuelve. Con la finalidad de poder gestionar los archivos JSON se hace uso del software libre proporcionado por Douglas Crockford, cuyo sitio web es el siguiente http://crockford.com/, en el sitio web http://www.json.org/Java/index.html, e incluido en el programa, para ser usado únicamente en este método. En cuanto a los citados archivos, se emplean los proporcionados por el tutor de proyecto Julio Villena Román, que contienen tweets de una temática particular en cada uno de ellos, lo que permite la fácil diferenciación posterior con la finalidad de comprobar los resultados, pero que no supone una ayuda al programa, ya que inmediatamente después de ser cargados todos ellos son juntados.

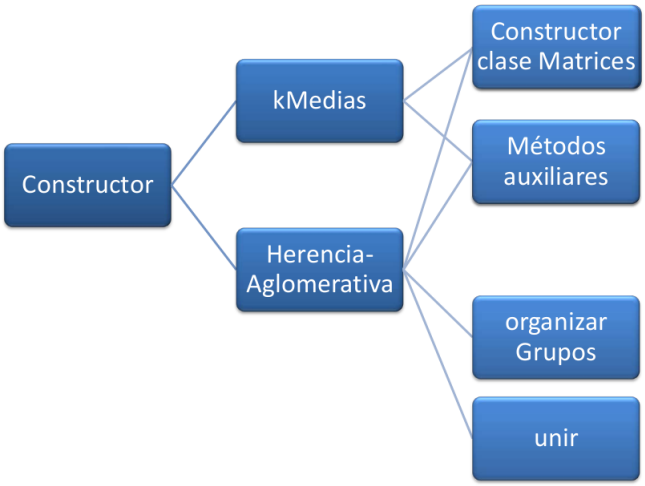

Figura 9: Diagrama de bloques clase Clustering

Clustering Esta clase, cuyo constructor es llamado por la clase Principal, se encarga de dividir los Strings recibidos en grupos por temática. Para ello no emplea directamente los propios Strings, sino que requiere que cada String sea convertido en un vector numérico, trabajo para el que a su

vez llama al constructor de la clase Matrices. Una vez que dispone de una matriz compuesta por los vectores de todos los Strings los agrupa, y posteriormente vuelve a convertir en los Strings originales. Ésta es la clase que implementa los dos algoritmos que han sido seleccionados, y que podría implementar todos aquellos que se deseen, sin realizar ningún tipo de cambio en las demás clases. Si bien no citaremos todos los métodos que esta clase contiene, puesto que la descripción sería excesiva y carente de interés, ya que cuenta con nueve métodos más uno constructor, sí que pasaremos a describir los más importantes. El primero de ellos es, naturalmente, el constructor, encargado de seleccionar, en función de los parámetros que se le transmitan, el algoritmo empleado para la selección, es decir, k-medias o herencia.

Los dos métodos más importantes son los que implementan cada uno de los dos algoritmos, siendo asistidos en esta labor por los demás métodos, en la mayoría de casos además siendo asistidos ambos métodos por los mismos. Dentro del método que implementa el algoritmo de k-medias remarcar el bucle que asigna, primero, a cada elemento al centroide más cercano, recalculando posteriormente el valor del centroide de cada grupo, y comprobando, por último, que se ha producido algún cambio, que evite que el bucle finalice. Dentro del método que implementa el algoritmo de herencia se ha decidido no incluir el código del bucle que realiza el funcionamiento en sí del algoritmo, sino que se ha decidido delegar esta función en otros dos métodos, denominados definirUniones y unir. El primero de ellos calcula cual es la distancia mínima entre dos centroides diferentes, e indica que han de unirse en grupos comunes todos los grupos que se encuentren entre sí a ésta. Téngase en cuenta que pueden ser varios, como sucede habitualmente por causa del concepto de retweet. El siguiente método se encarga de realizar las uniones que le ha indicado el método anterior, así como de recalcular el centroide de cada uno de los nuevos grupos generados. Una vez llamados ambos métodos se recalcula el número de grupos que quedan, dándose fin al bucle si y solo si el número de grupos resultantes es igual o menor al número indicado por el usuario.

Ha de tenerse en cuanta que si bien teóricamente puede darse el caso de que el número de grupos resultante sea menor al indicado, debido al hecho de que en cada iteración cabe la posibilidad de que se realicen múltiples uniones, ya sea en un mismo grupo de destino o en varios, este caso no acostumbra a darse. Esto se debe a que las uniones múltiples se ha comprobado empíricamente que tienden a darse en las primeras iteraciones, debido a que en este caso se unen todos los tweets que son iguales, o prácticamente, entre sí. Una vez unidos todos los tweets con diferencias ínfimas entre si, comienza la unión de los que se diferencian más, aunque pertenezcan a la misma temática, momento en el que las distancias entre cada dupla de grupos tienden a ser diferentes entre sí.

Por último, en ambos casos se genera el array de Strings a partir de las matrices de números obtenidas, siendo éste el valor devuelto por los método que implementan ambos algoritmos.

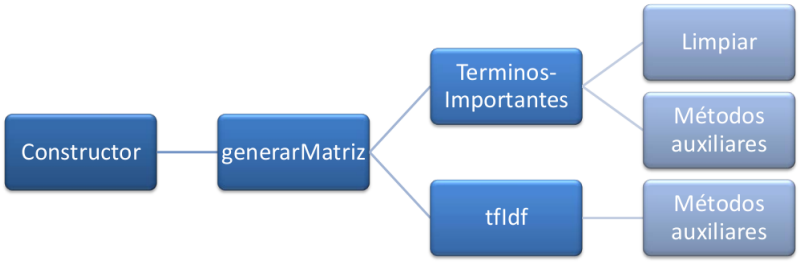

Figura 10: Diagrama de bloques clase Matrices

Matrices Esta clase se encarga de obtener la matriz numérica asociada a cada array de Strings. Su constructor es llamado por los métodos de la clase Clustering que implementan ambos algoritmos. Un objeto de la clase Matrices no contiene solamente la citada matriz, sino también el array que contiene la lista de términos que han sido tenidos en cuenta para generar la matriz. En total esta clase contiene un constructor, así como ocho métodos, por lo que no los describiremos todos, sino solo los que tienen más interés. El funcionamiento de esta clase se basa en el encadenamiento, es decir, el constructor llama a un método, que a su vez llama a otro, y así sucesivamente hasta que todos los metodos de la clase están involucrados, y se vuelve a retroceder hasta que el constructor obtiene la matriz numérica. Uno de los métodos a destacar es el método llamado limpiar. Su función es precisamente limpiar el texto de todo aquello que complique el agrupamiento, entendiendase mayúsculas, números así como todos los símbolos distintos de las letras, y también todas aquellas palabras que no aportan valor al agrupamiento, como pueden ser, de forma no excluyente, preposiciones, artículos y pronombres. Remarcar que éste es el único método de todo el programa en el que se tiene en cuenta el idioma que se indica a la clase Principal. La necesidad de este método de conocer el idioma en el que se encuentran los Strings proviene del hecho de que las citadas palabras que no aportan valor al agrupamiento son diferentes, como es lógico, en cada idioma. Con la finalidad de simplificar la futura implementación de nuevos idiomas se ha delegado la contención de la lista de palabras denominadas como "prohibidas" en un archivo de texto exterior al programa, conteniéndose en el programa solo el nombre del fichero. De este modo, creando un nuevo archivo de texto con las palabras "prohibidas" en otro idioma y sustituyéndose en la clase principal el valor idioma actual por la dirección del nuevo archivo , se permitiría sin ningún tipo de problema el uso del programa en ese nuevo idioma.
Otro de los métodos a destacar es el denominado "terminosImportantes". Su función radica en determinar que términos han de ser tenidos en cuenta. Para ello emplea la versión "limpiada" de los textos y de los términos, y comprueba en cuantos documentos aparece cada término.

Después comprueba si ese porcentaje de textos en los que aparece cada término pertenece al intervalo de porcentajes determinado por el usuario, y solo en caso de que sea así selecciona el término. Es en este método en el que se realiza el calibrado, consistente en comprobar empíricamente el porcentaje de error derivado del uso de cada intervalo, y seleccionar el que menor error produzca.
El tercer método que ha de ser tenido en cuenta se llama "tfidf", que no es otra cosa que las siglas en inglés de "Term frequency- Inverse document frequency" lo que significa frecuencia de término-frecuencia inversa de documento. La función de este método es calcular el valor de la relación entre cada término y cada documento. Para ello se calcula la frecuencia de un término en un documento, es decir la proporción de las veces que aparece un término en un documento frente al total de palabras que tiene el documento, y por otra parte la frecuencia inversa de documento, es decir, el total de documentos dividido por uno más el número de documentos que contiene la citada palabra, por último se multiplican ambos elementos. Se muestra en forma matemática para que resulte más sencillo de comprender:

$$tfidf = \frac{rep}{N_{palabras}} \times \frac{N_{doc}}{1 + N_{doc.palabras}} \tag{29}$$

Donde rep es el número de veces que aparece la palabra en el documento en cuestión, $N_{palabras}$ es el número total de palabras que tiene el texto, N_{doc} es el número total de documentos a agrupar, y $N_{doc.palabras}$ es el número total de documentos que contienen la palabra.
Al número de documentos que contienen la palabra se le suma un uno para evitar que se tome un valor excesivo en el caso de que la palabra sea contenida por solo ese documento, o incluso infinito en el caso de no ser contenida por ningún documento, caso este último que no podría darse en nuestro programa puesto que seleccionamos palabras que aparecen en los textos, pero que aumenta la robustez del programa ante posibles problemas.

13. Filtrado de los textos

Con la finalidad de evitar que muchas palabras que no son relevantes para el análisis, las denominadas palabras de parada o "stopwords", se conviertan en ruido en el análisis de los tweets se realiza una limpieza de éstos. Esta limpieza tiene una segunda finalidad, y es la de evitar que una misma palabra pueda ser considerada como diferente por llevar, o no, tilde o por comenzar, o no, con mayúscula.

El primer paso de la limpieza consiste en la transformación a minúscula de todas las letras, independientemente de que en un origen fueran mayúsculas o minúsculas. El segundo paso consiste en la eliminación de los símbolos añadidos a las letras, como pueden ser las tildes, las diéresis, o los múltiples acentos de los

que dispone un teclado (áàâ') y de la cecilla de la c (ç). El tercer paso consiste en la eliminación de todos aquellos símbolos diferentes de las letras del alfabeto español. Este paso fue determinado como bastante importante a raiz de que se detectara que muchos de los términos a partir de los cuales se estaba realizando el análisis eran emoticonos, independientes de la temática del tweet.

El cuarto paso es el encargado de las palabras de parada propiamiente dichas. Consiste en la eliminación de todas aquellas palabras que no aportan a la temática, como puedan ser, de forma no excluyente, artículos o pronombres.

14. Análisis de la frecuencia de término

Se ha realizado un análisis del porcentaje de documentos en los que aparece cada término, ordenándose los términos de menor número de documentos en los que aparecen a mayor. Se han dado cuatro casos, el caso de un solo archivo de entrada, es decir, una temática común para todos los documentos, el caso de dos archivos y temáticas, el de tres y el de cuatro. En la componente horizontal se muestra el orden de las palabras citado, y en la componente vertical se muestra el número de documentos en el que aparecen. Nótese que el total de documentos a analizar, que es el producto de 2000 por el número de archivos de cada análisis, es el máximo valor mostrado en cada una de las componentes verticales.

Inicialmente observamos la gráfica completa, en la que se puede observar que tiene la misma forma macroscópica en los cuatro casos, lo que nos indica una muy baja dependencia de la forma de la gráfica respecto al número de elementos. Esto nos va a permitir realizar una calibración independiente de este número, siempre que consigamos abstraernos de los valores absolutos de palabras, que crecen al crecer el número de documentos, aunque no de forma proporcional, sino con incrementos positivos en todos los casos, pero decrecientes y siempre que los posteriores aumentos de la gráfica mantengan esa similitud entre sí.

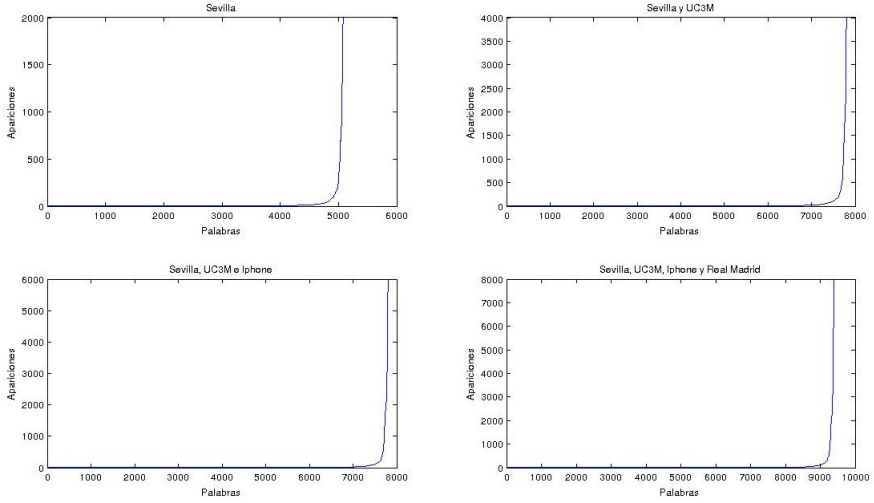

Figura 11: Relación entre las palabras y el número de documentos en las que aparecen.

Podemos observar que la gran mayoría de las palabras solo aparece en un conjunto muy pequeño de documentos cada una de ellas, mientras que una pequeña parte aparece en la gran mayoría. Puesto que tratamos de observar relaciones entre grupos de documentos, el conjunto que realmente nos interesa es este último, es decir, la parte izquierda de cada gráfica. Puesto que apenas podemos observar su forma, realizamos un aumento de las gráficas anteriores quedándonos con el 10 % más alto de valores, para observar con detalle la subida de la parte superior de la gráfica.

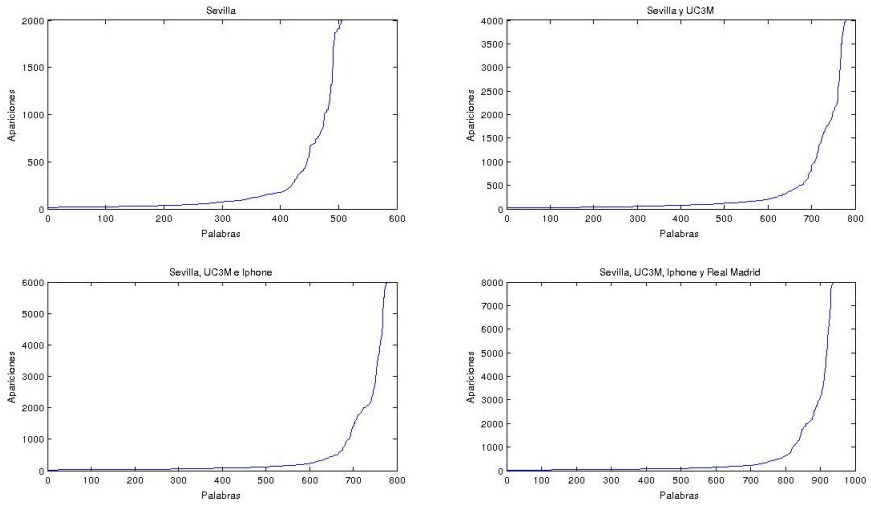

Figura 12: 10 % de palabras que aparecen en más documentos.

Podemos observar que si bien han aparecido ligeras diferencias entre las cuatro gráficas, todas siguen respondiendo al mismo formato. Podemos ver, también, que aunque ya el incremento sea fácilmente percibible en toda la gráfica, el gran aumento sigue produciéndose en una parte reducida de esta selección. Un buen ejemplo es la primera gráfica, donde se analizan poco más de 500 palabras, que aparecen en un máximo de 2000 documentos. puede observarse que hasta las 400 primeras palabras analizadas a penas se alcanza los 200 documentos cubiertos por cada palabra, lo que representa un alcance ligeramente inferior al 10 % del total. Por el contrario, en las poco más de 100 palabras restantes se cubre el intervalo entre el 10 % y el 100 % por completo.

Por último, pasamos a analizar la gran subida, es decir, el pequeño subconjunto de palabras que cubre la gran mayoría del intervalo de documentos en los que aparecer. Con esta finalidad mostramos otro aumento de la gráfica inicial, en este caso cubriéndose solamente el 1 % de palabras que aparecen en más documentos.

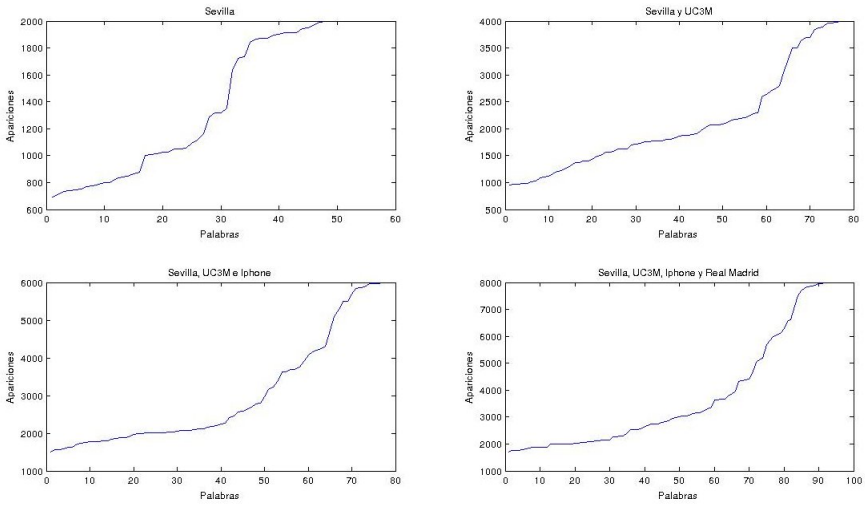

Figura 13: 1 % de palabras que aparecen en más documentos.

En esta imagen por fin observamos la parte que cubre el gran aumento con detalle. Se puede observar que aparecen diferencias entre las cuatro gráficas mostradas, pero estas diferencias se van reduciendo al aumentar el número de documentos empleados. Sí que se puede observar un cierto desplazamiento dependiente del número de documentos, resultando en que al aumentar el número de documentos el porcentaje de palabras que cubren la mayoría de los documentos disminuye.

Este hecho nos lleva a buscar un método de calibración que no dependa del porcentaje de palabras empleadas. Es por ello que se ha decidido a emplear el porcentaje de documentos cubiertos por cada palabra, en vez de el porcentaje de palabras en sí. Se multiplica el número total de documentos por dos valores, obteniéndose los límites superior e inferior. Es tras esta decisión técnica donde entra en escena la calibración, la calibración consiste en la comprobación empírica de cuales son los valores que deben tener ambos valores para minimizar el porcentaje de error.

15. Mejoras aplicadas al programa

15.1. Limpieza

La principal mejora aplicada al programa ha sido la simplificación de la extensión de los idiomas sobre los cuales puede trabajar, tratándose de que se pueda añadir cualquier idioma nuevo con las mínimas molestias posibles. Una dificultad asociada a la limpieza es que requiere de una lista de palabras que no son importantes, lo que a su vez requiere de la información de en que idioma se encuentran los tweets que recibe. De hecho es la única parte de todo el programa que ha de ser modificada al variar el idioma, dato que se ha de introducir en la clase Principal, y que va siendo transmitida hasta el método limpiar de la clase Matrices. Se ha buscado, en todo caso, que este problema asociado a la citada mejora resulte lo menos molesto posible para el usuario, a la vez que la implementación de cualquier otro idioma resulte lo más sencilla posible. Es por eso que se ha decidido delegar la lista de palabras de parada a un archivo aparte, de modo que no se tenga que buscar nada dentro del código con la excepción de la dirección de archivo al que acceder. En cuanto a la citada dirección se ha buscado también la sencillez, siendo ésta la razón por la que se transmite desde la clase Principal el dato, en vez de tener que introducirse en el propio método en el que va a ser utilizada. Al introducirse el parámetro junto a los demás parámetros a introducir, resulta más sencillo ser localizado.

16. Comparación de ámbos algoritmos

En esta sección se analizan cada una de las dos soluciones planteadas, comparando sus ventajas y desvantajas.

16.1. K-medias

El primer algoritmo planteado es el de k-medias. Éste es un algoritmo basado en centroides, cuyo funcionamiento consiste en la generación de varios centroides, tantos como grupos se deseen obtener, y la asociación a cada centroide de los elementos más cercanos según una distancia en particular. Posteriormente se ha de volver a generar el centroide como media de los elementos de cada grupo, y producir una re asignación, de modo cíclico hasta que en uno de los pasos ningún elemento cambie de grupo, dándose la situación en ese momento como resultado final. La distancia elegida para determinar la cercania de un elemento a un grupo es la distancia euclídea elevada al cuadrado, debido a la naturaleza de valores reales positivos, o cero, y truncados de los elementos del vector que representa la relación de cada documento con cada uno de los términos seleccionados. El cuadrado se ha elegido por que no supone ningún cambio en los resultados, pero simplifica operacionalmente los cálculos.

En el algoritmo de k-medias se ha de decidir el método por el se obtendrán las semillas, que serán los centroides de la primera iteración. Para esta solución se plantean e implementan dos opciones, la primera de ellas es la solución determinista, consistente en distribuir de forma uniforme cada semilla equidistante de las demás en cada una de las dimensiones. La segunda solución que se plantea es la solución aleatoria, consistente en determinar la posición de cada semilla en cada dimensión de forma aleatoria e independiente de su posición en las demás dimensiones, así como de la posición de las demás semillas en todas las dimensiones, con la única restricción de que dos semillas no pueden encontrarse en el mismo punto, teniendo en cuenta todo el conjunto de dimensiones, debido al hecho de que estas semillas serían imposibles de separar por las iteraciones, al encontrarse equidistantes de los elementos, dando lugar a uno grupo que contuviera a todos los elementos y otro, u otros, a ninguno, debido a las propias características de la implementación realizada. Otras implementaciones podrían dar lugar a dos, o más grupos que se repartieran los elementos de forma aleatoria, o bien asignándose el mismo número de elementos a cada uno de los dos o más grupos, o incluso a un algoritmo que no llegara a converger. En cualquiera de los casos la solución obtenida, de haberla, distaría mucho de ser satisfactoria, por lo que se bloquea esta opción.

Aunque se permite al usuario seleccionar ambas opciones, para los estudios de la calidad del algoritmo se ha seleccionado la opción de las semillas centradas, debido a que su determinismo permite comparar los resultados con mayor garantía. Esto se debe a que al no haber aleatoriedad en la selección de las semillas, unos mismos elementos de entrada producirán siempre un mismo resultado, garantizándose de este modo que cuando se realice alguna variación, el cambio en la salida será solo el proveniente de ese cambio y no la suma del

proveniente de ese cambio y el azar.

16.2. Herencia

La segunda solución planteada es el algoritmo de herencia aglomerativa con el método de los centroides. Se optó por el método de los centroides pues que resulta equivalente al método de la distancia media, como se demuestra en la subsección herencia de la sección Estado del arte, resultando computacionalmente mucho más eficiente. Frente a las opciones del vecino más próximo y del vecino más lejano, se trata de una opción intermedia, que evita el encadenamiento producido por el vecino más próximo sin dar lugar a grupos tan compactos como en el caso del vecino más lejano.

Frente a los métodos de herencia basados en división se decidió optar por un método aglomerativo debido a que en el caso habitual de que se deseara obtener un número pequeño de grupos se realizaría todo el reparto en un pequeño número de pasos, lo que implicaría un gran riesgo de que el resultado final estuviera muy dañado por errores producidos en un paso individual. Tampoco se ha optado por métodos basados en distribuciones estadísticas, puesto que no se cuenta con el dato de a qué distribuciones pueden pertenecer los elementos, aunque en la mayoría de casos se suele optar por distribuciones gausianas, y no se ha decidido correr ese riesgo. Además, en el caso de las distribuciones, así como en el caso de las densidades, si se desea obtener un resultado no demasiado forzado se ha de aceptar el número de grupos que el algoritmo determine, no cumpliéndose por tanto con el requerimiento de que se pueda determinar por parte del usuario el número de grupos. En cuanto a optar por otro sistema basado en centroides se consideraba que resultaba menos enriquecedor para el estudio el desarrollo de dos sistemas basados en el mismo concepto. Entiéndase que si bien este sistema emplea centroides no es un sistema basado en centroides, sino un sistema basado en herencia.

En cuanto a la estructura técnica de este algoritmo se emplea toda la estructura del algoritmo de k-medias con la excepción del método kMeans de la clase Clustering, que se ve sustituido por el método herenciaAglomerativa, reutilizándose, además, la mayoría de los métodos a los que llama kMeans. Debido a la reutilización se emplea también la misma distancia, que es la cuadrática de la distancia euclídea, así como la misma selección de términos, realizándose una calibración particular, puesto que puede darse que el intervalo óptimo no sea el mismo para todos los algoritmos.

16.3. Calibrado

El análisis descrito acontinuación sirve, por una parte, para optimizar el resultado de cada algoritmo, y por otra, para comparar los resultados ofrecidos por cada uno de ellos. Lo que se busca es obtener los valores superior e inferior entre los que se requiere que se encuentre una palabra para ser empleada en la creación del vector de valores asociado a cada tweet. Se realizan como siguen.

El error se determina como sigue, cada grupo de salida se asocia al grupo de entrada del que tenga más tweets, considerándose como errores todos aquellos tweets del grupo de salida que pertenezcan a cualquier otro grupo, una vez

obtenidos todos los errores se dividen por el número total de tweets empleados, obteniéndose la tasa de error que emplearemos para medir la calidad del resultado.

En primer lugar se realiza el calibrado grueso, consistente en ir calculando el resultado con cada valor superior e inferior, a saltos del 10 % excepto en el intervalo (0 %,5 %) en el que se realizan a saltos del 1 % puesto que la gran mayoría de los términos se encuentran ahí. Se analizan los resultados buscando la menor tasa de error posible.

Una vez obtenido el intervalo de los analizados que da lugar a una menor tasa de error se realiza el calibrado fino, que sigue el mismo procedimiento que el grueso, pero acotado a los valores que rodean a cada uno de los dos valores obtenidos en el calibrado grueso, y a saltos del 1 %.

Una vez obtenido el intervalo de calibración con un margen del 10 %, se realiza el calibrado fino, consistente en calcular el resultado obtenido con cada intervalo, pero no de todos los posibles, sino limitando la búsqueda al entorno de cada uno de los valores obtenidos en el calibrado grueso.

De este modo se obtienen los valores mayor y menor a los que debe ajustarse cada algoritmo.

16.4. Análisis del error

En esta sección mostraremos el resultado de aplicar el programa a un pequeño grupo de tweets, pero mostrando los propios tweets, y no los resultados numéricos como hasta ahora. En ambos casos se realizarán las mismas pruebas, consistentes en introducir tweets de dos temáticas en el primer caso, y de tres en el segundo, de forma que se puedan comparar los resultados.

16.4.1. K-medias

En primer lugar realizamos la prueba con dos grupos de tres tweets cada uno de ellos, que serán el grupo denominado "Madrid" y el grupo denominado "Qashqai", al programa se le indica que devuelva dos grupos, dando lugar a lo siguiente:

```
Grupo llamado: bueno******************************
1 1673658370 "Todos dan favorito al Barça y eso es bueno":
Carlo Ancelotti, entrenador del Real Madrid, dijo sobre el
Clási... http://t.co/1EJrhCEObu
2 2059546624 "Todos dan favorito al Barça y eso es bueno" http://t.co/mLbR1Z1mez #LigaE
El grupo 0 tiene 2 clusters diferentes.
Grupo llamado: mala*******************************
0 2127056896 Abro mi cartera y m encuentro un tiket d compra
en el Paseo d la Castellana en Madrid dl 20-10-2013 y no voy
ha Madrid dsd 1999... Tú dirás!
0 -1681784832 @luchitowilliams @tuconcierto @ManuelCorredera
@eloypincay @TiresExpressPma No sería mala unos rines para
```

```
el Qashqai jejeje!!! Saludos!
1 -1345847296 Vaya mañana. Agua agua y más agua.Tenía que
haber comprado el qashqai versión anfibio con aletas y timón
2 -452427776 #cocheselectricos Nissan Qashqai 1.5 DCI 110CV
DPF ACENTA en Málaga de ocasion Diesel con...
http://t.co/kSbSQG6UyF
El grupo 1 tiene 4 clusters diferentes.
```

Puede observarse que solo ha habido un tweet que está en el grupo erroneo, que sería el primer tweet del segundo grupo. A pesar de hablar sobre la ciudad de Madrid se ha introducido en un grupo en el que todos los demás tweets hablan sobre el Qashqai.

En segundo lugar realizamos la prueba con tres grupos de dos tweets cada uno de ellos, que serán el grupo denominado "Real Madrid", el grupo denominado "Qashqai" y el grupo denominado "Sevilla", al programa se le indica que devuelva tres grupos, dando lugar a los siguientes resultados:

```
Grupo llamado: real*****************************
0 -301436928 RT @real_pasion: Barcelona - Real Madrid
#Horarios #ClasicoMundial http://t.co/GmWK6y1hqD
1 -800571392 RT @real_pasion: Barcelona - Real Madrid
#Horarios #ClasicoMundial http://t.co/GmWK6y1hqD
El grupo 0 tiene 2 clusters diferentes. Grupo llamado: sevilla*************************
0 -1681784832 @luchitowilliams @tuconcierto @ManuelCorredera
@eloypincay @TiresExpressPma No sería mala unos rines para
el Qashqai jejeje!!! Saludos!
0 -1882714112 RT @orgullobiri: El Málaga está dispuesto a
fichar un delantero del Sevilla http://t.co/9RPbBexVsj
#sevillafc
1 -1123540992 @cristobalsoria yo en la vida privada de la
gente no me meto, viva el sevilla y a seguir dándole caña
a los vikingos
El grupo 1 tiene 3 clusters diferentes. Grupo llamado: qashqai************************
1 -1345847296 Vaya mañana. Agua agua y más agua.Tenía que
haber comprado el qashqai versión anfibio con aletas y timón
El grupo 2 tiene 1 clusters diferentes.
```

Volvemos a encontrarnos con un tweet fuera de sitio, el primero del segundo grupo, que debería estar en el tercer grupo, puesto que es donde se encuentra el otro tweet que trata sobre el "Qashqai" y ninguno más.

16.4.2. Herencia

En primer lugar realizamos la prueba con dos grupos de tres tweets cada uno de ellos, que serán el grupo denominado "Madrid" y el grupo denominado "Qashqai", al programa se le indica que devuelva dos grupos y da los siguientes resultados:

```
Grupo llamado: es****************************
0 2127056896 Abro mi cartera y m encuentro un tiket
d compra en el Paseo d la Castellana en Madrid dl
20-10-2013 y no voy ha Madrid dsd 1999... Tú dirás!
0 -1681784832 @luchitowilliams @tuconcierto @ManuelCorredera
@eloypincay @TiresExpressPma No sería mala unos rines
para el Qashqai jejeje!!! Saludos!
1 1673658370 "Todos dan favorito al Barça y eso es
bueno": Carlo Ancelotti, entrenador del Real Madrid,
dijo sobre el Clási... http://t.co/1EJrhCEObu
2 2059546624 "Todos dan favorito al Barça y eso es
bueno" http://t.co/mLbR1Z1mez #LigaBBVA
2 -452427776 #cocheselectricos Nissan Qashqai 1.5 DCI
110CV DPF ACENTA en Málaga de ocasion Diesel con...
http://t.co/kSbSQG6UyF
El grupo 0 tiene 5 clusters diferentes.
Grupo llamado: qashqai****************************
1 -1345847296 Vaya mañana. Agua agua y más agua.Tenía
que haber comprado el qashqai versión anfibio con aletas
y timón
El grupo 1 tiene 1 clusters diferentes.
```

Podemos observar como el programa, en vez de crear dos grupos de iguales o similares tamaños ha creado un grupo con un solo tweet y otro con todos los demás.

En segundo lugar realizamos la prueba con tres grupos de dos tweets cada uno de ellos, que serán el grupo denominado "Real Madrid", el grupo denominado "Qashqai" y el grupo denominado "Sevilla", al programa se le indica que devuelva tres grupos, dando lugar a los siguientes resultados:

```
Grupo llamado: http****************************
0 -301436928 RT @real_pasion: Barcelona - Real Madrid
#Horarios #ClasicoMundial http://t.co/GmWK6y1hqD
0 -1882714112 RT @orgullobiri: El Málaga está dispuesto
a fichar un delantero del Sevilla http://t.co/9RPbBexVsj
#sevillafc
1 -800571392 RT @real_pasion: Barcelona - Real Madrid
#Horarios #ClasicoMundial http://t.co/GmWK6y1hqD
2 -1190629376 RT @futbol11_frases: "¿Miedo al Real Madrid?
El barrio donde yo vivía si que daba miedo que solo había
traficante de drogas"  -Tévez. http:...
2 -452427776 #cocheselectricos Nissan Qashqai 1.5 DCI 110CV
DPF ACENTA en Málaga de ocasion Diesel con...
http://t.co/kSbSQG6UyF
El grupo 0 tiene 5 clusters diferentes.
Grupo llamado: rt****************************
```

```
0 -1681784832 @luchitowilliams @tuconcierto @ManuelCorredera
@eloypincay @TiresExpressPma No sería mala unos rines para
el Qashqai jejeje!!! Saludos!
2 -435683328 RT @SevillaInsolita: Las murallas de Sevilla
comenzaron a derribarse a partir de la revolución de 1868
debido al crecimiento de la ciudad. ...
El grupo 1 tiene 2 clusters diferentes.
Grupo llamado: http****************************
1 -1345847296 Vaya mañana. Agua agua y más agua.Tenía que
haber comprado el qashqai versión anfibio con aletas y timón
1 -1123540992 @cristobalsoria yo en la vida privada de la
gente no me meto, viva el sevilla y a seguir dándole caña a
los vikingos
El grupo 2 tiene 2 clusters diferentes.
```

En el primer grupo se puede observar una predominancia de tweets sobre el Real Madrid, a pesar de lo cual hay un tweet de cada uno de los otros dos grupos. En el segundo grupo hay solo dos tweets, siendo éstos de diferentes grupos de origen. Lectura similar podemos hacer en el tercer grupo, coincidiendo incluso los grupos de origen con los del segundo grupo, "Sevilla" y "Qashqai".

16.5. Puntos fuertes de cada algoritmo

16.5.1. K-medias

La principal ventaja del algoritmo de k-medias ha sido su error, muy inferior al algoritmo de herencia, no solo en la calibración, sino también, una vez calibrados ambos, en el estudio del error que se ha mostrado en el punto anterior, en el que se puede observar con claridad como los resultados ofrecidos por el algoritmo de k-medias superan notablemente a los resultados ofrecidos por el algoritmo de herencia.

Otra ventaja muy a tener en cuenta del algoritmo de k-medias es el tiempo de ejecución, notablemente inferior al del algoritmo de herencia. Se han ejecutado ambos algoritmos en se han evaluado en un portátil Intel® Core™ i3 CPU M 380 @ 2.53GHz × 4 con 3,7 GB de RAM y microprocesador de 64 bits, con el sistema operativo Ubuntu 12.04 LTS de 64 bits. Si bien este tiempo de ejecución depende del propio calibrado citado, sí que puede ser acotado entre dos límites, dependiendo de si se le requiere que emplee todos los términos, o bien una parte muy reducida de ellos. Con 4000 tweets estos valores son de 1 minuto y 29 segundos en el caso de que se le requieran emplear todos los términos, lo que alcanza un total de 7752, y de 16 segundos en el caso de que se le requiera emplear un solo término. En el caso estimado como óptimo por la calibración el tiempo de ejecución alcanza los 17 segundos. Para ese mismo caso de calibración el tiempo empleado por el algoritmo de herencia es de 5 minutos y 37 segundos.

En cuanto al tiempo de ejecución en el algoritmo de k-medias ha de tenerse en cuenta que depende, en una medida bastante equivalente entre sí, de tres factores, el número de documentos que se le introducen, el número de términos que se emplean y el número de grupos que se desean obtener. Ha de tenerse en cuenta, eso sí, que el número de elementos que se le introducen afecta al número de términos que se han de tomar en cuenta, por lo que el efecto de este elemento ha de ser muy tenido en cuenta.

16.5.2. Herencia

La principal ventaja con la que cuenta este algoritmo frente al de k-medias es que no cuenta con su dependencia frente al número de grupos que se le requieren. Este algoritmo va repitiendo el bucle mientras reduce el número de grupos, por lo que tener que devolver un número mayor de grupos solo le requiere detener el algoritmo antes, reduciéndose el tiempo que el método que aplica el algoritmo requiere. El tiempo total de ejecución del programa se ve compensado por un mayor tiempo requerido al mostrar los resultados, pero en cualquier caso se ve solucionado el problema de la alta dependencia respecto a este concepto.

Otra ventaja potencial, puesto que no es aprovechada por el programa, es que podría mostrarse en una misma ejecución diferentes números de grupos, puesto que el programa ha de recorrer los números más altos como paso intermedio para llegar al valor requerido. De este modo el usuario podría contar por una parte con los resultados simplificados que da el tener un pequeño número de grupos, pero podría acceder a subdivisiones menos importantes en el caso de requerirlo accediendo a pasos anteriores sin necesitar realizar otra ejecución del programa, con el correspondiente ahorro de tiempo. Ventaja que en cualquier caso se ve notablemente reducida por el efecto de uno de los problemas que se muestran a continuación.

16.6. Problemas de cada algoritmo

16.6.1. K-medias

El primero de los problemas a citar fue detectado en el método de la clase Matrices que selecciona los términos que han de ser tenidos en cuenta al analizar los términos que más veces aparecían con diferentes conjuntos de tweets. detectándose que el único término que aparecía siempre en el 100 % de los casos era lo que podríamos denominar término nulo, que es el que no contiene nada, entiéndase esto “”, y que es asignado a todos los documentos. El empleo de este término hace que el algoritmo no converja, de modo que el programa nunca finaliza.

Este problema no tiene efectos en la práctica, puesto que al ser un término que siempre aparece en todos los documentos solo acaba formando parte de los términos a tener en cuenta de seleccionarse como límite superior de porcentaje de documentos en los que puede aparecer un término el 100 %, opción que cuando menos nunca puede ser considerada la única mejor, ya que un término que

aparece en todos los documentos resulta de muy poca ayuda en la discriminación entre estos. No es cierto que no aporte nada, puesto que el valor que tiene relacionado con cada documento no tiene porque ser el mismo ya que esto dependería de la fórmula empleada para calcular el vector, pero en cualquier caso resulta de poca ayuda. Además, el calibrado tiende a dejar los valores extremos, es decir, cercanos al 0 % y al 100 %, fuera. Como solución a este error, se aconseja realizar los estudios de calibrado tomando como límite superior 99.

El segundo problema a tener en cuenta es la alta dependencia que el algoritmo K-means tiene respecto al número de grupos que se pueden generar. Como se ha indicado, se le ha de decir al programa el número de grupos que se desean obtener a la salida. En caso de aumentar este concepto, el tiempo de ejecución aumenta, lo que puede suponer un problema en el caso de que se desee emplear el programa para crear muchos grupos pequeños, que con otro sistema no habrían requerido tanto tiempo.

16.6.2. Herencia

Este algoritmo cuenta con dos problemas graves, pasamos a describir cada uno de los dos problemas más detalladamente:

El principal problema con el que cuenta este algoritmo es su error, derivado del resultado de generar un grupo muy grande, convirtiendo a los demás grupos en residuales. Este efecto puede observarse en los dos casos de prueba que se han realizado en el apartado de análisis del error. Es un efecto bastante dañino ya que da lugar a que una parte importante de los elementos de la mayoría de los grupos siga unida entre sí en el grupo grande.

El segundo problema a tener en cuenta es el tiempo de ejecución. Mientras el algoritmo de K-means tarda aproximadamente 17 segundos en dividir 4000 tweets en dos grupos, el algoritmo de herencia requiere de 5 minutos y 37 segundos. Además la distancia temporal entre las ejecuciones de ambos métodos se incrementa al aumentar el número de tweets introducido, como demuestra el hecho de que para dividir 6000 tweets en dos grupos el algoritmo de k-medias requiere de 26 segundos, mientras que el algoritmo de herencia requiere de 8 minutos y 49 segundos. Ha de tenerse en cuenta que el tiempo de ejecución no solo supone un problema en desarrollo al aumentar el tiempo de trabajo derivado de mejoras, así como de la propia calibración ya citada, aumentando por tanto los costes de producción, sino que también supone un gran problema en cuanto a conseguir aceptación por parte de los usuarios finales. Esta última situación se deriva de que un usuario requiere de muchas más razones, o cuando menos de razones más contundentes, para utilizar un programa cuando éste le hace esperar durante un largo tiempo, declinándose por opciones más rápidas, aunque éstas sean ligeramente inferiores en calidad, en cuanto le resulte posible, con la finalidad de deshacerse de esos incómodos tiempos de espera. Otro problema añadido a los largos tiempos de ejecución es la imposibilidad de emplearlos como aplicaciones secundarias de otras aplicaciones, puesto que superan el tiempo de ejecución esperado por éstas, con lo que una aplicación en paralelo de ambas aplicaciones con la finalidad de mostrar los resultados a la vez al final

requeriría de que la otra aplicación tenga que esperar a ésta, convirtiéndose nuestra aplicación en el problema.

16.7. Conclusiones de la comparación

Una vez analizados ámbos algoritmos se ha elegido la solución k-medias, debido, por una parte a su inferior tasa de error, y por otra a su menor tiempo de ejecución. Ésta será por tanto la solución que se plantée como final.

Parte IV
Análisis a posteriori de la solución obtenida

En esta sección se tratará de realizar una evaluación del resultado final obtenido en el programa, por lo que solo se observará la opción finalmente elegida, es decir, la opción del algoritmo k-medias con una calibración que le lleva a emplear las palabras que se encuentran en entre el 48 % y el 54 % de los documentos, puesto que esa combinación es el programa final.

17. Puntos fuertes

A continuación se muestran las principales ventajas del programa.

17.1. Facilidad de implementación de nuevos idiomas

Como ya se ha citado anteriormente, el programa requiere de información dependiente del idioma en el que se le introducen los tweets que habrá de dividir, para la correcta selección de las palabras que son importantes y las que no. Si bien esto requiere trabajo extra cada vez que se desea emplear un nuevo idioma, se ha tratado por todos los medios de reducir las dificultades que esto requiere al usuario. Y se puede considerar un objetivo bastante alcanzado, puesto que solamente se ha de modificar un parámetro, que además se encuentra junto a los demás parámetros que pueden ser modificados por parte del usuario, para que el programa acceda al nuevo archivo de palabras de parada. Además en este archivo solo se requiere que se introduzca cada una de las palabras que se desean eliminar en una linea diferente, sin ningún tipo de simbología especial.

En cualquier caso añadir que el programa funcionará también sin disponer del archivo particular del idioma en el que se encuentran los tweets, con el defecto, eso sí, de que entonces el efecto de la limpieza en la mejora de la calidad del resultado se verá notablemente reducido, ya que se limitará a la primera parte, es decir, la eliminación de simbología especial. Este problema reducirá la calidad puesto que introducirá palabras que no aportan sobre la temática, pero no más allá del funcionamiento de un algoritmo que no contenga esta selección de palabras importantes. Se estaría simplemente anulando parcialmente una de las mejoras implementadas.

18. Problemas

18.1. Casos plurilingües

Este problema se basa en el hecho de que en casos de tweets creados en diferentes idiomas el programa encontraría muchas diferencias entre tweets de la misma temática por el simple hecho de que todos los términos que dependen del idioma serán diferentes entre ellos. Ha de entenderse que la diferencia no es absoluta, un tweet que trate, por ejemplo, sobre el Real Madrid podrá contener palabras como Real, Madrid, Casillas o Florentino independientemente de que esté escrito en español, inglés o alemán, pero las palabras fútbol, gol o equipo ya no van a ser comunes a todos ellos, creando una separación entre cada uno de los tres que realmente no debería existir, puesto que la temática es la misma.

Podría temerse que a este problema se añada el de la diferencia de los archivos que contienen las palabras de parada según el idioma, pero en este caso sí que nos encontramos ante un problema que tiene una fácil solución. La creación de un fichero que sea la unión de los ficheros de cada uno de los idiomas de los tweets contenidos en esa búsqueda sería una solución bastante aceptable. Eso sí, no sería una solución carente de todo riesgo, puesto que en algunos casos se da la coincidencia de que palabras que en un idioma deben ser eliminadas, en otro son palabras que contienen mucho significado, dos ejemplos claros serían las preposiciones alemanas "nach" (hacia) y "mit" (con), que deberían aparecer en el fichero correspondiente de alemán, supondrían la eliminación de los términos "Nach" y "M.I.T." (recuérdese que las diferencias de escritura son eliminadas en la limpieza) que son, respectivamente, un cantante de hip hop alicantino y una universidad politécnica de Massachusetts, Estados Unidos. A pesar de lo cual, en la gran mayoría de casos estas combinaciones no implican ningún problema.

En cualquier caso ha de tenerse en cuenta que toda minería de textos que no cuente con un traductor va a pecar de diferencia entre palabras en diferentes idiomas que tengan el mismo significado, puesto que un programa es incapaz de trabajar con significado, sino que trabaja con cadenas de Strings, siendo capaz solamente de medir diferencias entre éstas, de forma booleana, o de forma cuantitativa. Estas diferencias serán muy altas en el caso de dos palabras de idiomas diferentes, sobre todo en el caso de que ambos lenguajes no compartan origen.

18.2. Temáticas de amplitud diferente

Esta situación se refiere al caso en el que en el conjunto de tweets estudiados se encuentren grupos con la misma temática interna, y diferente temática externa, es decir grupos que deberían ser mostrados separados a la salida, pero de una gran diferencia entre el número de sus tweets. Este problema se deriva del hecho de que los grupos grandes van a imponer la mayoría de sus palabras más empleadas, relegando a los grupos pequeños a contar con pocas palabras propias, o en casos extremos ninguna, en la lista de palabras a tener en cuenta.

Se realiza una demostración en la que se cuenta con dos grupos de 2000

tweets cada uno, frente a un tercer grupo que solo cuenta con 137 tweets, o lo que es lo mismo, un 6,85 % del tamaño de cada uno de los otros. Se ajusta el número de grupos de salida a 3, y el análisis de los grupos generados da lugar al siguiente código, en el que ha de entenderse que los grupos de entrada son los grupos que introducimos al programa, y de los que ya conocemos su temática, y los grupos de salida son los generados por el programa a partir de su análisis de la temática:

Grupo 0, el que contiene 137 tweets:

```
El grupo 0 de entrada tiene 5 valores en común con el
grupo 0 de salida
El grupo 0 de entrada tiene 1 valores en común con el
grupo 1 de salida
El grupo 0 de entrada tiene 131 valores en común con el
grupo 2 de salida
```

Grupo 1, uno de los que contiene 2000 tweets:

```
El grupo 1 de entrada tiene 181 valores en común con el
grupo 0 de salida
El grupo 1 de entrada tiene 1335 valores en común con el
grupo 1 de salida
El grupo 1 de entrada tiene 484 valores en común con el
grupo 2 de salida
```

Grupo 2, uno de los que contiene 2000 tweets:

```
El grupo 2 de entrada tiene 357 valores en común con el
grupo 0 de salida
El grupo 2 de entrada tiene 41 valores en común con el
grupo 1 de salida
El grupo 2 de entrada tiene 1602 valores en común con el
grupo 2 de salida
```

Puede observarse que si bien la gran mayoría de los tweets del grupo problemático, el grupo 0 de entrada, se encuentran alojados en el mismo grupo de salida, el 2, este grupo es el correspondiente al grupo dos de entrada, puesto que el 72,25 % de los tweets de este grupo de salida se corresponden con él. Por tanto podemos deducir que los tweets del grupo pequeño han quedado ocultos en un grupo que habría de asociarse a otra temática.

Este problema tiene difícil solución, ya que el conocimiento del origen de los tweets es solo planteable en un caso de prueba, puesto que en un caso real es lo que deseamos obtener del programa, y por tanto, con lo que no contamos, ya que de contar con ello carecería de sentido el uso del programa. Al no contarse con la diferenciación a priori entre los grupos grandes y pequeños no se puede realizar una discriminación que nos coloque los términos de los grupos pequeños entre los valorados, puesto que no resultan diferenciables de los términos que aparecen en una pequeña parte de los otros grupos más grandes.

Parte V
Conclusiones y trabajos futuros

19. Conclusiones

La realización de este proyecto ha supuesto un reto en dos aspectos diferenciados entre sí, aunque también relacionados. En primer lugar el acercamiento a un mundo que a mí (alumno) me resultaba desconocido, como es el de la minería de datos. He descubierto un entorno nuevo, pero notablemente necesario en la sociedad actual debido a la globalización de las comunicaciones y, por ende, al acceso desde cualquier punto del mundo a todas las redes de la información de acceso público, entiéndase que sigue habiendo información de acceso privado. En cualquier caso, nos encontramos ante una sociedad en la que tenemos más información de la que podemos gestionar, por lo que necesitamos la ayuda de las máquinas para tal labor, y ninguna máquina mejor que la que nos ha ayudado a obtener esa misma información. El mundo de la minería de datos tiene otro atractivo especial, debido a que a partir de comportamientos puramente mecánicos, como son todos los realizados por un ordenador, se dan resultados que los seres humanos consideramos como humana, al menos subjetivamente puesto que los análisis creados con la finalidad de comprobar la inteligencia humana de un interlocutor son de mayor dureza que la opinión subjetiva de un ser humano. Un buen ejemplo sería el caso de este proyecto, un programa que, simplemente, analizando cadenas de strings, da como resultado la división por lo que entendemos como temática del tema. El acercamiento al tema, me ha resultado por tanto tan complejo como interesante, al resultarme completamente nuevo.

El segundo aspecto en el que ha habido un reto, es la propia implementación del proyecto. Esto es, la programación en sí. El desarrollo de un proyecto de programación completo desde cero requiere más trabajo que los ejercicios realizados en grupos, o los desarrollos de métodos separados, en el aspecto de escritura del código, y ya no digamos, en la propia estructuración, es decir, en la decisión de dividir las actividades en un conjunto u otro de clases, y dentro de cada una de ellas en un cierto número u otro de métodos. De hecho en el segundo caso ha habido necesidades de realizar cambios, como en el caso de la opción finalmente rechazada de herencia, en el que se trataba inicialmente de incluir todo el código que realiza el propio trabajo indicado por el algoritmo teórico en el método principal. Finalmente se consideró que esto suponía una problemática enorme en cuanto a comprensión del código por parte de seres humanos, lo que llevó a dividir este trabajo no ya en un método externo sino en dos, simplificando bastante la comprensión. En resumen, un gran aporte al conocimiento subjetivo sobre la división en métodos y clases de un programa.

Por todo ello, valorar muy positivamente el aprendizaje obtenido en el campo de la minería de datos, así como en el lenguaje de programación Java.

20. Ideas para el futuro

La primera idea que nos viene a la cabeza para el futuro es el desarrollo de una aplicación capaz de gestionar más idiomas. Como se ha comentado, la aplicación permite la implementación sencilla de más idiomas, lo que simplifica la internacionalización del producto. Pero lo que se está planteando aquí es un paso más. Una vez implementados un conjunto de idiomas, los que sean siempre que sean más de uno, se podría realizar un análisis de los términos del conjunto de tweets introducidos, con la finalidad de determinar a que idioma pertenecen. Para ello se realizaría una búsqueda de las propias palabras de parada de cada idioma en todos los documentos, de modo que se seleccionase el idioma con mayor aparición de palabras de parada entre los documentos. Esta mejora simplificaría aún más la experiencia del usuario, al no tener que indicar el idioma en el que está realizando la búsqueda. La aplicación estaría limitada, según esta estructura, a que los tweets introducidos estuvieran todos en el mismo idioma, puesto que realizar una selección individualizada de los tweets podría dar lugar a los problemas anteriormente citados para el estadio actual de desarrollo.

La segunda idea que resultaría interesante realizar con la aplicación es la implementación del mismo concepto a un motor de búsqueda, ya sea de páginas web, de documentos dentro de una base de datos, etc. La misma problemática que se plantea en el caso de los tweets se plantea también en los otros casos citados, puesto que también se realizan búsquedas en las que los mismos términos puedan tratar de temáticas diferentes dependiendo del caso, produciéndose el mismo problema de ruido que en el caso de los tweets. Debería considerarse, eso sí, mayores tiempos de respuesta, por estarse trabajando con textos mucho más extensos, en la mayoría de los casos. También requeriría una gestión de la información añadida en cuanto a extracción de los datos de la web se refiere.

Una tercera opción, ya sea a partir de la estructura actual, o aplicándose a la segunda idea ya citada, es la de la implementación de la aplicación a modo de plugin en un explorador, que permitiera emplear la aplicación al mismo tiempo que se realizan búsquedas ya sea en Twitter, en el primero de los casos, o en un explorador o buscador de documentos en el caso de la segunda idea, de modo automático, como si su funcionalidad formara parte del servicio web del que se está haciendo uso.

La cuarta idea es la de aprovechar la estructura ya creada para la implementación de más algoritmos diferentes a los actuales, de modo que éstos puedan ser estudiados como lo han sido las dos soluciones planteadas.

Parte VI
Historia del proyecto

21. Estructura del documento

A continuación se mostrará la estructura por capítulos del presente trabajo:

Presentación En la primera parte se presenta el trabajo que se expondrá en partes posteriores, buscando mostrar al lector la motivación que ha llevado a realizarlo, así como aportando la información básica para que el lector sepa identificar el tema que se va a tratar. Por último se muestran los objetivos que se satisfarán en temas posteriores.

Estado del arte En esta sección se realiza un análisis detallado de la situación actual de la ciencia del clustering de modo que, por una parte, se realice un acercamiento documentado a esta ciencia y, por otra,se puedan conocer las mejores posibilidades de cara al desarrollo posterior. Para estructurar el análisis se muestran inicialmente los conceptos que han de conocerse en profundidad para comprender el clustering, y posteriormente se desglosan las diferentes formas de concebir el clustering.

Estrategias en el desarrollo de la solución Aquí se muestra la estrategia empleada en el desarrollo de la solución obtenida. Para ello, se describe inicialmente la estructuración técnica del programa. Posteriormente se detallan las mejoras aplicadas al programa con la finalidad de mejorarlo. Se realiza también un análisis de la legalidad vigente con la finalidad de comprobar que se esté respetando, y por último se analizan y comparan las dos soluciones planteadas.

Análisis a posteriori de la solución obtenida En esta sección realizamos un análisis de la solución una vez finalizada ésta, tratando de valorar los resultados, tanto los más positivos como los más negativos.

Conclusiones y trabajos futuros Posteriormente a la finalización del desarrollo, se muestran las conclusiones finales obtenidas de éste, así como los posibles avances planteados para trabajos futuros.

Historia del proyecto Aquí se analiza el trabajo realizado en cuanto a estructura, planificación de la gestión del tiempo así como costes totales derivados de éste.

22. Planificación

El proyecto se basa en la investigación de los algoritmos de agrupamiento automático de datos y posterior desarrollo de una solución derivada que permita la separación de tweets por su temática. Por tanto, se ha de realizar una división

entre las dos partes del proyecto, siendo la primera la investigación, y la segunda el desarrollo técnico, con las correspondientes subdivisiones según mostraremos a continuación:

Investigación Consiste en la búsqueda de información teórica sobre la temática del clustering, con la finalidad de conocer la situación actual de la ciencia de la minería de datos, así como de las posibilidades que ésta aporta para el posterior desarrollo.

Estrategias Análisis de los requerimientos técnicos de la solución a implementar, así como de las posibilidades prácticas para satisfacerlos.

Implementación Escritura del programa, eliminación de problemas técnicos, aplicación de mejoras al algoritmo, optimización de la ejecución, y análisis a posteriori del resultado.

Documentación elaboración de la memoria que expone todo el proceso.

En el siguiente diagrama de Gantt se muestra el reparto en días del trabajo realizado. Ha de tenerse en cuenta que la dedicación en horas no es constante respecto a la cantidad de días, dándose días de dedicación completa, hasta días en los que no se ha realizado ninguna tarea, pasando por días de dedicación parcial en los que se ha compaginado el desarrollo del proyecto con alguna otra tarea no relacionada con éste.

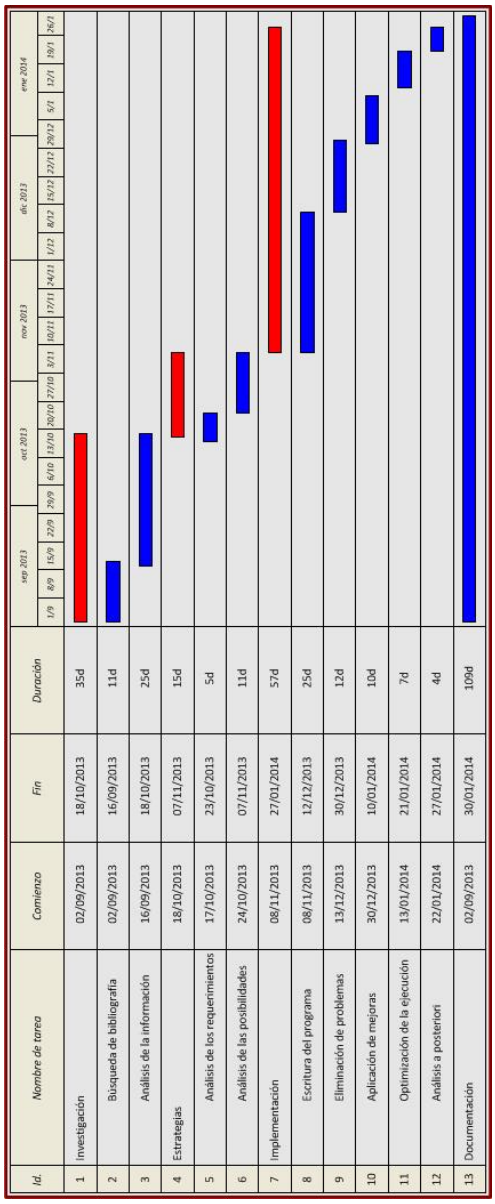

Figura 14: Diagrama de Gantt que representa las diferentes tareas del proyecto distribuidas en el tiempo.

23. Presupuesto

El presupuesto se divide en costes directos e indirectos, los indirectos se estimarán como el 10 % de los directos, y los directos se mostrarán a continuación.

23.1. Costes directos

Los costes directos se dividen a su vez en costes de personal y costes de amortización, mostramos cada uno de ellos por separado.

23.1.1. Costes de personal

En el desarrollo del proyecto se han visto involucrados dos perfiles, el del tutor, que será considerado como ingeniero senior, y el del alumno, que será considerado como ingeniero junior, con las siguientes cargas temporales.

Concepto	Horas Ingeniero Senior	Horas Ingeniero Junior	TOTAL
Búsqueda de bibliografía	5	35	40
Análisis de la información	5	55	60
Análisis de los requerimientos	5	20	25
Análisis de las posibilidades	10	30	40
Escritura del programa	0	60	60
Eliminación de problemas	5	30	35
Aplicación de mejoras	10	35	45
Optimización de la ejecución	0	30	30
Análisis a posteriori	10	20	30
Documentación	10	35	45
TOTAL	60	350	410

Cuadro 2: Costes de personal asociados al proyecto.

Un trabajador tiene 4 semanas de vacaciones al año lo que supone 20 días de ocio añadido a los fines de semana, a los que hemos de sumar los días de fiesta, que según el calendario laboral de la Comunidad de Madrid para el año 2013 ascendieron a 14 días, por lo que quedan 331 días. Teniendo en cuenta que durante un año promedio hay 104 días que caen en fin de semana, nos quedan 227 días en los que un trabajador trabaja al año según la ley. Aplicando una jornada completa de 8 horas, contaríamos con 1816 horas de trabajo anual.

Se estiman unos sueldos de mercado de 40 000€ anuales para el ingeniero senior y de 22 000€ anuales para el ingeniero junior. A ambos sueldos se les ha de sumar una aportación a la seguridad social del 30 %, obteniéndose un total de 52 000€ anuales para el ingeniero senior, y de 28 600€ anuales para el ingeniero junior. Dividiendo por el total de horas calculadas al año se obtienen unos costes por hora de 28,64€ para el ingeniero senior y de 15,75€ para el ingeniero junior. Con estos costes por hora calculamos el coste laboral total.

Perfil	Coste por hora	Número de horas	Coste total
Ingeniero senior	28,64€	60	1718,4€
Ingeniero junior	15,75€	350	5512,5€
TOTAL			7230,9€

Cuadro 3: Costes laborales del proyecto.

23.1.2. Costes de amortización

Para la realización del proyecto se ha contado con los equipos informáticos del ingeniero senior y del ingeniero junior. A los ordenadores se les estima una amortización de 5 años, es decir, 60 meses. Hemos de tener en cuenta que el proyecto ha durado 5 meses. Pasamos a desglosar los costes correspondientes:

Ordenador	Precio	Coste de amortización
Portatil Ingeniero Senior	450€	52,5€
Ordenador de Sobremesa Ingeniero Junior	400€	46,67€
TOTAL		99,17€

Cuadro 4: Costes de amortización

23.2. Costes totales

Los costes indirectos se estiman como el 10 % de los costes directos, lo que nos dará unos costes totales como sigue:

Concepto	Valor
Costes directos	7330,07€
Costes indirectos	733,01€
TOTAL	8063,08€

Cuadro 5: Costes totales del proyecto.

Hemos de concluir por tanto que el coste total del proyecto asciende a la cifra de 8063,08€.

Leganés a 30 de Enero del 2014

Fernando España García

Referencias

[1] Autor: Caron. Título: J."Moore's law", Editorial: Tele.com, vol. 4, no. 12, pp. 54.

[2] Autores: Brian S. Everitt; Sabine Landau; Morven Leese; Daniel Stahl. Título: Cluster Analysis, 5th Edition Editorial: John Wiley & Sons

[3] Autor: Russell K. Anderson. Título: Visual Data Mining: The VisMiner Approach, 2nd Edition

[4] Autor: Han, Jiawei. Título: Data mining: concepts and techniques. Editorial: Elsevier Science

[5] Autores: Foster Provost; Tom Fawcett. Título: Data Science for Business. Editorial: O'Reilly Media, Inc

[6] Autores: Elena Deza, Michel-Marie Deza. Título: Dictionary of Distances. Editorial: Elsevier B.V.a

[7] Autores: Ronen Feldman;James Sanger. Título:The Text Mining Handbook. Editorial: Cambridge University Press

[8] Autor: Sarah Boslaugh. Título: Statistics in a Nutshell, 2nd Edition. Editorial: O'Reilly Media, Inc.

[9] Título: "Ley Orgánica 15/1999, de 13 de diciembre, de protección de Datos de carácter personal" Fecha: versión consolidada el 5 de Marzo de 2011

[10] Empresa: Twitter. Título: API de Twitter. Dirección web: https://dev.Twitter.com/docs/API/1.1/overview